

205

/AN EXPLORATORY STUDY OF
SOFTWARE DEVELOPMENT MEASURES
ACROSS COBOL PROGRAMS/

by

NADINE M. VEEDER

B.S., KANSAS STATE UNIVERSITY, 1984

A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

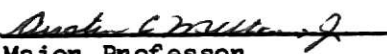
MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1986

Approved by:


Major Professor

LD
2668
.24
1986
V43
c. 2

Table of Contents

A11206 692181

Chapter	Page
1. Introduction	1
2. Analysis of the Data	4
2.1. The Data Collection	4
2.2. Measures Used to Analyze the Data Collection	5
2.3. Rational Behind Selected Measures	9
2.4. Tools Developed for Analyzing the Data	11
2.4.1. Preprocessing of Programs	13
2.4.1.1. Removal of Superfluous "White Space"	13
2.4.1.2. Distinguishing of Statements	13
2.4.1.3. Indentation of a Program	14
2.4.2. Difference Module	14
2.4.3. Tabulation	15
2.4.4. Report Production	16
2.4.5. Other Modules	16
2.5. Method of Analysis	16
3. Results of Analysis	19
3.0. Introduction	19
3.1. Program Characteristics	19
3.1.1. Number[TYPE] and Number[TYPE]/Number[ALL]	20
3.1.2. Number[TYPE,LEVEL]	21
3.1.3. ANL[TYPE]	21
3.2. Added Section Characteristics	22
3.2.1. add_sections[TYPE,LEVEL]	22
3.2.1.1. Entire Program Collection	22

3.2.1.2.	Programs in the Early Phases of	23
	Development	
3.2.1.3.	Programs in the Later Phases of	24
	Development	
3.2.2.	ANL_Add_Sections[TYPE] and	25
	SD_ANL_Add_Sections[TYPE]	
3.2.2.1.	Entire Program Collection	25
3.2.2.2.	Programs in the Early Phases of	25
	Development	
3.2.2.3.	Programs in the Later Phases of	26
	Development	
3.3.	Added Statement Characteristics	26
3.3.1.	Adds[TYPE], %Adds[TYPE], and SD_%Adds[TYPE] . . .	26
3.3.1.1.	Entire Program Collection	26
3.3.1.2.	Programs in the Early Phases of	27
	Development	
3.3.1.3.	Programs in the Later Phases of	28
	Development	
3.3.2.	adds[TYPE,LEVEL]	29
3.3.2.1.	Entire Program Collection	29
3.3.2.2.	Programs in the Early Phases of	29
	Development	
3.3.2.3.	Programs in the Later Phases of	30
	Development	
3.3.3.	ANL_Adds[TYPE] and SD_ANL_Adds[TYPE]	31
3.3.3.1.	Entire Program Collection	31

3.3.3.2.	Programs in the Early Phases of	31
	Development	
3.3.3.3.	Programs in the Later Phases of	30
	Development	
3.4.	Changed Statement Characteristics	33
3.4.1.	Changes[TYPE], Changes%(TYPE),	33
	%Changes[TYPE], and SD_%Changes[TYPE]	
3.4.1.1.	Entire Program Collection	33
3.4.1.2.	Programs in the Early Phases of	34
	Development	
3.4.1.3.	Programs in the Later Phases of	35
	Development	
3.4.2.	changes[TYPE,LEVEL]	36
3.4.2.1.	Entire Program Collection	36
3.4.2.2.	Programs in the Early Phases of	37
	Development	
3.4.2.3.	Programs in the Later Phases of	38
	Development	
3.4.3.	ANL_Changes[TYPE] and SD_ANL_Changes[TYPE] . . .	39
3.4.3.1.	Entire Program Collection	39
3.4.3.2.	Programs in the Early Phases of	40
	Development	
3.4.3.3.	Programs in the Later Phases of	40
	Development	
3.4.4.	CHANGES[TYPE1,TYPE2] and CHANGES%(TYPE1,TYPE2] .	41
3.4.5.	The Nesting Environment of Changed	42
	IF/ELSE Statements	

3.5. Deleted Statement Characteristics	43
3.5.1. Deletes[TYPE], Deletes%[TYPE],	44
%Deletes[TYPE], and SD_%Deletes[TYPE]	
3.5.1.1. Entire Program Collection	44
3.5.1.2. Programs in the Early Phases of	45
Development	
3.5.1.3. Programs in the Later Phases of	46
Development	
3.5.2. deletes[TYPE,LEVEL]	47
3.5.2.1. Entire Program Collection	47
3.5.2.2. Programs in the Early Phases of	48
Development	
3.5.2.3. Programs in the Later Phases of	48
Development	
3.5.3. ANL_Deletes[TYPE] and SD_ANL_Deletes[TYPE] . . .	49
3.5.3.1. Entire Program Collection	49
3.5.3.2. Programs in the Early Phases of	50
Development	
3.5.3.3. Programs in the Later Phases of	50
Development	
3.5.4. The Nesting Environment of Deleted	51
IF/ELSE Statements	
4. Conclusions and Suggestions for Future Work	53
References	61
Appendix	63

List of Figures

Figure	Page
1. Number of Revisions for programs	4
2. Hierarchy Diagram of the Tool Set Modules	12
3. Number[TYPE] and Number[TYPE]/Number[ALL]	63
for all programs	
4. Number[TYPE] and Number[TYPE]/Number[ALL]	64
for revision 0-2 programs	
5. Number[TYPE] and Number[TYPE]/Number[ALL]	65
for revision 6-11 programs	
6a. Number[TYPE,LEVEL] for all programs (Levels 0-2) .	66
6b. Number[TYPE,LEVEL] for all programs (Levels 3-5) .	67
7a. Number[TYPE,LEVEL] for revision 0-2 programs . . .	68
(Levels 0-2)	
7b. Number[TYPE,LEVEL] for revision 0-2 programs . . .	69
(Levels 3-5)	
8a. Number[TYPE,LEVEL] for revision 6-11 programs . . .	70
(Levels 0-2)	
8b. Number[TYPE,LEVEL] for revision 6-11 programs . . .	71
(Levels 3-5)	
9. ANL[TYPE]	72
10. add_section[TYPE,LEVEL] for all programs	73
11. add_section[TYPE,LEVEL] for revision 0-2 programs .	74
12. add_section[TYPE,LEVEL]	75
for revision 6-11 programs	

13.	ANL_Add_Sections[TYPE]	76
14.	SD_ANL_Add_Sections[TYPE]	77
15.	Adds[TYPE] and %Adds[TYPE] for all programs	78
16.	Adds[TYPE] and %Adds[TYPE] for revision 0-2 programs	79
17.	Adds[TYPE] and %Adds[TYPE] for revision 6-11 programs	80
18.	SD_%Adds[TYPE]	81
19.	adds[TYPE,LEVEL] for all programs	82
20.	adds[TYPE,LEVEL] for revision 0-2 programs	83
21.	adds[TYPE,LEVEL] for revision 6-11 programs	84
22.	ANL_Adds[TYPE]	85
23.	SD_ANL_Adds[TYPE]	86
24.	Changes[TYPE], %Changes[TYPE], and Changes%[TYPE] . for all programs	87
25.	Changes[TYPE], %Changes[TYPE], and Changes%[TYPE] . for revision 0-2 programs	88
26.	Changes[TYPE], %Changes[TYPE], and Changes%[TYPE] . for revision 6-11 programs	89
27.	SD_%Changes[TYPE]	90
28a.	changes[TYPE,LEVEL] for all programs (Levels 0-2) .	91
28b.	changes[TYPE,LEVEL] for all programs (Levels 3-5) .	92
29.	changes[TYPE,LEVEL] for revision 0-2 programs . . .	93
30a.	changes[TYPE,LEVEL] for revision 6-11 programs . . (Levels 0-2)	94
30b.	changes[TYPE,LEVEL] for revision 6-11 programs . . (Levels 3-5)	95

31.	ANL_Changes[TYPE]	96
32.	SD_ANL_Changes[TYPE]	97
33.	CHANGES[TYPE1,TYPE2] AND CHANGES%(TYPE1,TYPE2) . .	98
	for all programs	
34.	Deletes[TYPE], %Deletes[TYPE], and Deletes%(TYPE) .	100
	for all programs	
35.	Deletes[TYPE], %Deletes[TYPE], and Deletes%(TYPE) .	101
	for revision 0-2 programs	
36.	Deletes[TYPE], %Deletes[TYPE], and Deletes%(TYPE) .	102
	for revision 6-11 programs	
37.	SD_%Deletes[TYPE]	103
38a.	deletes[TYPE,LEVEL] for all programs (Levels 0-2) .	104
38b.	deletes[TYPE,LEVEL] for all programs (Levels 3-5) .	105
39.	deletes[TYPE,LEVEL] for revision 0-2 programs . . .	106
40.	deletes[TYPE,LEVEL] for revision 6-11 programs . .	107
41.	ANL_Deletes[TYPE]	108
42.	SD_ANL_Deletes[TYPE]	109
43.	Depth of nesting of IF statements	110
44.	Environment of changed IF statements	111
45.	Environment of changed ELSE statements	111
46.	Environment of deleted IF statements	112
47.	Environment of deleted ELSE statements	112
48.	Environment of changed IF statements	113
	(percentages)	
49.	Environment of changed ELSE statements	113
	(percentages)	

- 50. Environment of deleted IF statements 114
 (percentages)
- 51. Environment of deleted ELSE statements 114
 (percentages)
- 52. Environment of changed/deleted IF/ELSE statements . 115
 (percentages)

Acknowledgements

The author wishes to express her sincere gratitude to Dr. Austin C. Melton Jr., Dr. David A. Gustafson, and the Software Measures Seminar participants for their guidance and assistance during the course of this research. A special thanks is also extended to the corporation which most graciously supplied the data for this research effort.

The author also wishes to thank her parents Mr. and Mrs. Merton H. Veeder for their love and support through many years of education.

Chapter One

Introduction

The software development process is complex and difficult to understand. Many variables including the hardware available, the language used, the type of problem to be solved, the algorithm used in solving the problem, the methodology used in developing the software, the programming environment, and the programmer's ability affect the software development process. While many of these factors can be easily controlled, the variability of the human element is inherent. Each programmer will approach the task of software development in a different manner, and worse yet, some programmers may unnecessarily increase the complexity of the task at hand.

Software managers have recognized the need for objective methods for monitoring the progress of software development and for anticipating problems in the development process. Software complexity measures/metrics are an attempt to satisfy this need. The terms measures and metrics are used for the most part interchangeably in the literature. The term measure will be used in this report. A Software complexity measure, in this context, is a quantification of the attributes of a program. This study quantifies the attributes of modifications made to a program during development, as such the measures described in this

report are software development measures.

Several software complexity measures have been proposed and explored. In general, these measures can be classified as one of the following: size measures, data structure measures, logic structure measures, composite measures, design measures, cost measures, or defect measures. All of these measures are plagued with being environment dependent, requiring re-calibration of parameters when transported from one environment to another (ie. historical data). Since measures are environment dependent, empirical studies and experimentation are required to assess the effects of the environment on a measure.

Even though empirical studies and experimentation are essential for the development and assessment of a software measure, the cost of the data gathering process often prevents many studies and experiments from occurring. "Real world" data is especially difficult to obtain as industry is rarely willing to expend the time, effort, and dollars to acquire the necessary data. Therefore, many studies are conducted in a university environment under controlled situations. While these studies are certainly worthwhile and commendable, "real world" data is an essential ingredient in assessing measures which will be applied to "real world" software development.

The study described in the following chapters is an exploratory study of "real world" COBOL programs in the

development phase. This study generated voluminous amounts of data in order to better understand what goes on in the software development phase and also to gain some insight into the software development process in general.

Chapter two describes the data analyzed in this study as well as the measures applied to the data, the tools which implemented these measures, and the methodology used in analyzing the data.

Chapter three describes the results of this study. This description is broken into five categories: measures on all statements in the program collection, measures on the added sections in the program collection, measures on the added statements in the program collection, measures on the changed statements in the program collection, and measures on the deleted statements in the program collection.

Chapter four includes the conclusions of this work, along with suggestions for future work.

Chapter Two

Analysis of the Data

2.1. The Data Collection

Thirty one COBOL programs in the development phase were analyzed. These programs were obtained from a Kansas corporation, and thus were written by "professional" programmers. The programs were relatively short (on average 1630 lines of code), and had varying numbers of revisions (See Figure 1.) Hereafter, the term revision will be used to refer to a particular revision of a program.

revisions	number of programs
1	6
2	3
3	7
4	7
5	3
6	3
7	1
12	1

Figure 1. Number of Revisions for Programs

Since the data gathering process was not done in a controlled environment, several difficulties arose in analyzing

the data.

- 1) Little was known about the steps in the software life cycle preceding the coding phase.
- 2) Little was known about how revisions were gathered, and furthermore there was no assurance that all revisions labeled as revision 0 were actually the first attempt at coding the program. (Investigations of the programs provided evidence to the contrary.)
- 3) Program collections were incomplete, and few programs with large numbers of revisions existed.
- 4) Structured coding techniques (including modularization) were not utilized, and therefore module by module analysis was not possible.

2.2. Measures Used to Analyze the Data Collection

Two sequential revisions of a program were analyzed using the measures described below. The result of this analysis was the quantification of the modifications made to the first revision which yielded the second revision. The measures were computed using the Data Division statements and the Procedure Division statements only. The Identification Division and Environment Division statements were excluded from this analysis due to their relative stability.

(The notation used below was devised by Dr. David A. Gustafson and the participants of the Software Measures Seminar at Kansas State University.)

TYPE: represents the type of statement being analyzed where ALL represents the collection of all statement types

Statement types were classified by their functions. The classification used is described below. (Only those statement types occurring in the data are listed below.)

```
TYPE ::= ALL | comments | declarations |
        assignments | conditionals | branches |
        input/output | labels | other
comments ::= spacing purposes | textual
assignments ::= MOVE | ADD | SUBTRACT | COMPUTE
conditionals ::= IF | ELSE | ON | AT END
branches ::= CALL | PERFORM | GOTO | NEXT | EXIT
input/output ::= DELETE | DISPLAY | OPEN | READ |
               REWRITE | WRITE
other ::= EXAMINE | INSPECT | SEARCH | SORT | SET |
        EXEC CICS | GOBACK
```

Number[TYPE]: number of statements of specified TYPE in the program.

Changes[TYPE]: number of statements of specified TYPE that
have been changed.

Deletes[TYPE]: number of statements of specified TYPE that
have been deleted.

Adds[TYPE]: number of statements of specified TYPE that
have been added.

Add_Sections[TYPE]: number of statements of specified TYPE
after which a section of code has been
added.

Modification: as a notational convenience, let
$$\text{Modification} ::= \text{Changes} \mid \text{Deletes} \mid \text{Adds} \mid$$
$$\text{Add_Sections}.$$

It is assumed that the values of Modifica-
tion and TYPE are constant for an equation;
for example, Modification%[TYPE] may never
be Changes[IF]/Adds[ELSE].

Modification%[TYPE]: $\text{Modification[TYPE]} / \text{Number[TYPE]}$

%Modification[TYPE]: $\text{Modification[TYPE]} / \text{Modification[ALL]}$

SD_%Modification[TYPE]: standard deviation of
 $\% \text{Modification[TYPE]}$

ANL: average level of nesting for a program.

Let N_i be the number of statements at nesting level i

Then $\text{ANL} =$

$$(\text{summation } (i=0 \text{ to } N) \text{ of } i * N_i) / \text{Number}[\text{ALL}]$$

ANL[TYPE]: average level of nesting of the statements of
TYPE in a program.

Let N_i be the number of statements of TYPE at
nesting level i

Then ANL[TYPE] =

$$(\text{summation } (i=0 \text{ to } N) \text{ of } i * N_i) / \text{Number}[\text{TYPE}]$$

ANL_Modification[TYPE]: ANL of Modification[TYPE]

SD_ANL_Modification[TYPE]: standard deviation of
ANL_Modification[TYPE]

Number[TYPE, LEVEL]: number of statements of TYPE that are
at LEVEL of nesting.

changes[TYPE, LEVEL]: number of changes of statements of
TYPE that are at LEVEL of nesting.

deletes[TYPE, LEVEL]: number of deletions of statements of
TYPE that are at LEVEL of nesting.

adds[TYPE, LEVEL]: number of additions of statements of
TYPE that are at LEVEL of nesting

add_sections[TYPE, LEVEL]: number of statements of TYPE
after which a section of code
has been added that are at LEVEL
of nesting.

CHANGES[TYPE1, TYPE2]: number of changed statements of

TYPE1 which changed to a statement
of TYPE2

$CHANGES\%[TYPE1,TYPE2]: CHANGES[TYPE1,TYPE2]/Changes[TYPE1]$

In addition to the above measures, one additional software development characteristic was quantified. The nesting environment of a changed or deleted IF/ELSE statement was quantified.

2.3. Rational Behind Selected Measures

The measure $Modifications\%[TYPE]$ is the percentage of statements of a specified TYPE which are being modified relative to the total number of statements of that TYPE in the program. (Here $Modifications ::= Deletes \mid Changes$) This measure indicates the relative stability of a particular statement type in a program. The program size heavily influences this measure.

The measure $\%Modifications[TYPE]$ is independent of the program size. In the case of $Modification ::= Deletes \mid Changes$, this measure shows which statement types are being modified most frequently relative to all modified statements. In the case of $Modification ::= Adds$, this measure shows which statement types are being added most frequently. In the case of $Modification ::= Add_Sections$, this measure depicts the most likely places for sections of code to be added.

The measure SD_%Modification[TYPE] reflects the dispersion of values for %Modification[TYPE] from the mean. If the dispersion is small and thus the numbers are grouped closely together, the measure of %Modification[TYPE] is more reliable and descriptive of the data, than if the spread is greater.

The measure ANL[TYPE] depicts the average nesting level of a statement type. This measure indicates where a particular statement type is most likely to be located.

The measure ANL_Modification[TYPE] depicts the average nesting level of statement types which are modified. This measure can indicate at which level modifications are most likely to occur.

The measure SD_ANL_Modification[TYPE] reflects the dispersion of values for ANL_Modification[TYPE].

The measures changes[TYPE,LEVEL], deletes[TYPE,LEVEL], adds[TYPE,LEVEL], and add_sections[TYPE,LEVEL] show the actual number of statements of a particular type which are modified at each level. This measure gives a clearer picture of where statements are being modified than the previous measures.

The measures CHANGES[TYPE1,TYPE2] and CHANGES%[TYPE1,TYPE2] depict the number of statements of TYPE1 which are changed to a statement of TYPE2, and the percentage of statements of TYPE1 which are changed to a

statement of TYPE2, respectively. It would appear that statements would change to either a statement of the same type or a statement of a similar type. These measures were applied to the programs in order to determine if this were the case.

The environment of a changed or deleted statement was quantified to determine if the environment of a modified statement influenced the amount of modifications occurring. Highly nested code is generally believed to add to the complexity of the source code. Intuitively, as the complexity of the source code increases, so should the number of modifications occurring. Thus, it would most likely be the case that statements in the encompassing scope of a highly nested statement will change quite frequently. To serve as a test case, the IF and ELSE statements were considered in this analysis. Even though only two statements were used in this analysis, it should be the case that any statement in the encompassing scope of a highly nested statement will change quite frequently.

2.4. Tools Developed for Analyzing the Data

A set of tools was developed to analyze the COBOL programs. This set includes preprocessing modules, a difference module, tabulation modules, and report producing modules. Figure 2 gives a hierarchy diagram of the tool set modules.

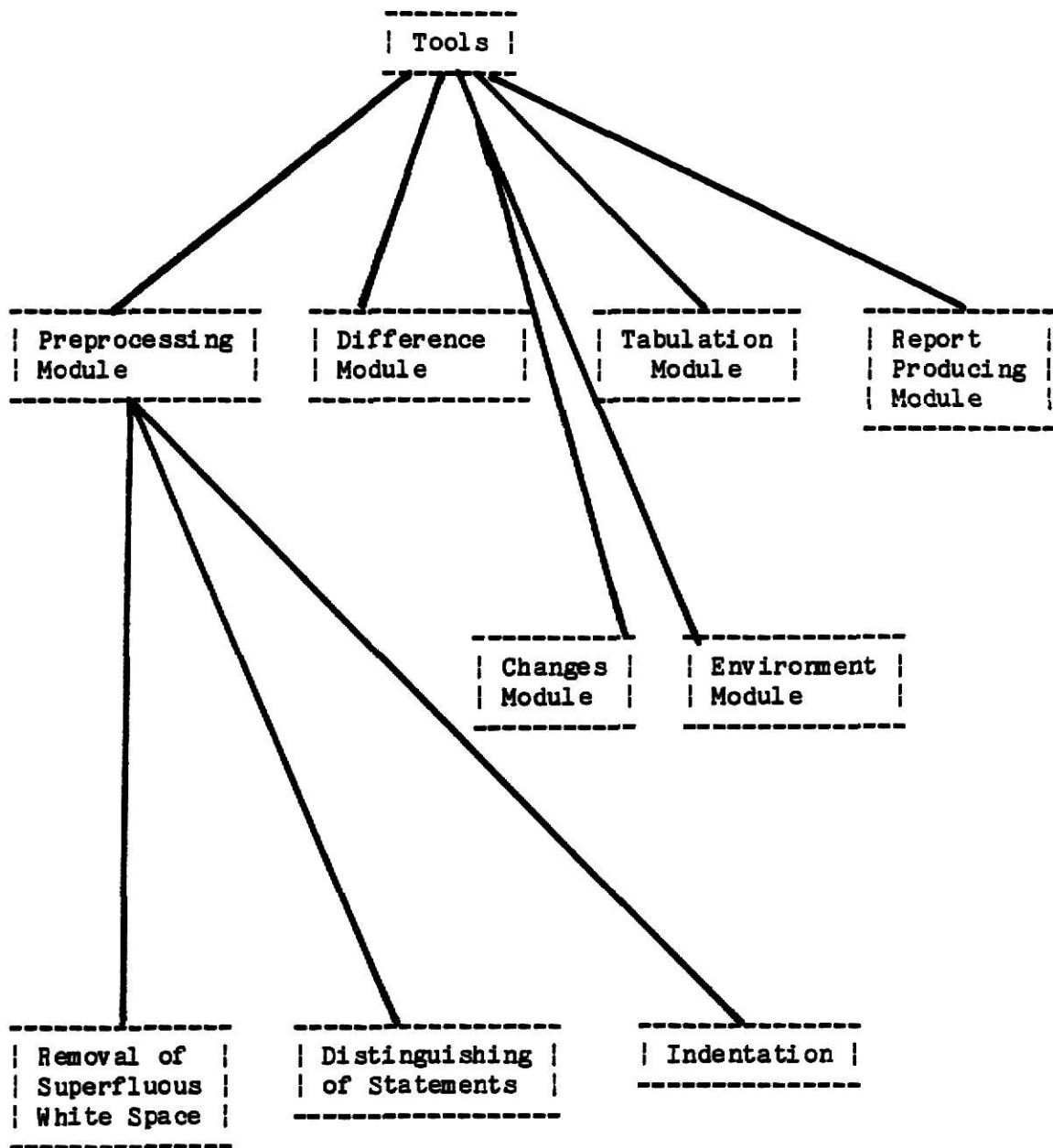


Figure 2. Hierachy Diagram of the Tool Set Modules

2.4.1. Preprocessing of Programs

Preprocessing of the programs was necessary to prepare them for the difference module. The preprocessing modules have three basic functions: 1) to remove all superfluous "white space" in the programs, 2) to make distinctions between similar statements in different Divisions of a COBOL program, and 3) to ensure uniform indentation.

2.4.1.1. Removal of Superfluous "White Space"

All superfluous blanks, tabs, and newlines were removed to ensure proper identification of a change. For example, if the only change between Statement A in revision one and revision two is the addition of a blank or a tab, Statement A has not been significantly changed. Therefore, this type of a change was ignored. Similarly, if a statement is spread over two lines in revision one and occupies three lines in revision two but otherwise is the same, no significant change has occurred. This type of change was ignored by ensuring that each statement occupies only one line.

2.4.1.2. Distinguishing of Statements

Initially, each program was searched to verify the existence of all four Divisions. (Identification Division, Environment Division, Data Division, and Procedure Division) Special flags were inserted into the programs where these

Divisions were missing. Using either the actual Divisions or the special flags as reference points, statements belonging to each of the divisions were prefixed by a unique character signifying which Division a particular statement belonged to. The prefixing of statements by unique division characters was done to ensure that each statement, when out of its context, could be properly identified. For example, if the statement "MOVE X TO Y." appeared in the Identification Division as a Remark, it should not be considered as a MOVE statement but rather as a comment. However, had preprocessing of this "MOVE" statement not occurred, later modules would not be able to make this distinction.

2.4.1.3. Indentation of a Program

A standard indentation scheme was imposed upon the programs. The statements were indented one tab character per each level with level 0, the outermost level, having no indentation. Due to the unstructuredness of the COBOL programs, only IF statements were used to determine the nesting level.

2.4.2. Difference Module

A difference module, utilizing the "diff" utility, had been developed and used previously for analyzing changes to maintenance programs written in C. This module, with minor

alterations, was used to analyze the COBOL programs.

This module compares two programs and notes changes, deletions, and added sections by prefixing statements in the first revision by special characters. This module can also be used to obtain which statements were added to the first revision by interchanging the inputs to the module.

2.4.3. Tabulation

Using the results of the difference module, four additional files are created. One file consisting of all the changed statements, one consisting of all deleted statements, one consisting of all the added statements, and one file consisting of all the statements after which a section of code was added. Each of these four files along with the entire program is used in sequence as input to the main tabulation module.

The main tabulation module counts the number of each type of statement and the number of each statement type at each level of indentation. These counts are written to a temporary file.

Upon completion of this module, five new files exist: an overall counts file, a changed statement counts file, a deleted statement counts file, an added statement counts file, and an added section counts file.

2.4.4. Report Production

Several short C and Shell programs were written to produce several different reports from the temporary files created by the tabulation module. One report was generated for each of the measures described in Section 2.2.

2.4.5. Other Modules

Two additional tools were written which utilize the output of the difference tool. One module utilizes the "diff" utility to analyze the types of statements which changed statements were changed to (ie., the resulting type of changed statements). The other module utilizes the "awk" utility to examine the environment of a changed statement.

2.5. Method of Analysis

The initial objective of this analysis was to be able to discern distinct patterns from revision to revision in program development. However, due to the problems discussed in Section 2.1, this objective could not be fully realized. Thus, the following approach in analyzing the data was adopted.

- 1) All programs were analyzed using the measures described in Section 1.2.
- 2) Programs in early phases of development were

analyzed using the measures described in Section 1.2.

- 3) Programs in the later phases of development were analyzed using the measures described in Section 1.2.

This approach allowed the characteristics of the entire program collection, along with the characteristics of programs in the early phases and later phases of development to be analyzed. The intent of this approach was to compare the characteristics of programs in the early phases of development with the characteristics of programs in the later phases of development, thus allowing distinctions to be made between the characteristics of programs in the early phases of revision and programs in the later phases of revision.

Programs in revisions 0 through 2 inclusive were selected as programs in the early phases of development. Programs having 6 or more revisions were selected as programs in the later phases of development. A wider variance in revision numbers were included as programs in the later phases of development to allow the collection to be more representative of the entire program collection. Unfortunately, only one program had more than 7 revisions, while 4 programs had 6 or more revisions. (See Figure 1.)

Two sets of programs were excluded from this analysis. One program set was excluded on the basis that it had no

modifications through the entire development cycle. The second program was excluded due to radical changes during development, this program doubled or tripled in size with each revision.

Chapter 3

Results of Analysis

3.0. Introduction

The results from the analysis of the data collection will be presented as follows:

- 1) characteristics of the programs,
- 2) characteristics of the added sections in programs,
- 3) characteristics of the added statements in programs,
- 4) characteristics of the deleted statements in programs,
- 5) characteristics of the changed statements in programs.

3.1. Program Characteristics

The program collection as a whole will be discussed in this section. The characteristics of programs in the early phases of development and later phases of development are not presented in this section; however, the characteristics may be observed in the Appendix in Figures 3-9. The characteristics of programs in the early phases of revisions and programs in the later phases of revision are not presented

as they are not necessarily indicative of any patterns of development, but rather were used as a reference point for other measurements which were applied to the data.

3.1.1. Number[TYPE] and Number[TYPE]/Number[ALL]

The entire program collection had a total of 157,575 statements in the Data Division and Procedure Division combined. 33.64% of the statements were assignment statements; 23.27% of the statements were declarations; 13.82% of the statements were conditional statements; 11.77% of the statements were comments; 10.93% of the statements were branching statements; 3.85% of the statements were labels; 2.42% of the statements were other statements; and 0.30% of the statements were input/output statements. (See Appendix Figure 3.)

The relatively high percentage of comments is rather misleading. 69.21% of the comments were used only for vertical spacing of the code. 30.79% of the comments were "textual" comments. These were comments containing some text. Unfortunately with rare exceptions, these were statements being commented out for debugging purposes.

In the category of branching statements, the unstructuredness of the code is revealed in the relatively high percentage of GOTO statements. 6.23% of all the statements were GOTO statements (9813) and 3.22% of the statements were PERFORM statements (5067). (See Appendix Figure 3.)

3.1.2. Number[TYPE,LEVEL]

The level of nesting of statements ranged from level 0 to level 7. 50.12% of the statements were at level 0; 36.49% of the statements were at level 1; 10.92% of the statements were at level 2; 1.63% of the statements were at level 3; 0.44% of the statements were at level 4; 0.19% of the statements were at level 5; 0.19% of the statements were at level 6; and 0.07% of the statements were at level 7.

Levels 0 through 2 inclusive had a wide variety of statement types. However, as the nesting level increased, the variety of statements present decreased. At level 3, eight different statement types were present, they were textual comments, MOVE statements, IF statements, ELSE statements, PERFORM statements, GOTO statements, NEXT statements, and OPEN statements. However, at level 7, only three different statement types were present: textual comments, MOVE statements, and PERFORM statements. (See Appendix Figures 6a and 6b.)

3.1.3. ANL[TYPE]

The average nesting level of the entire program collection was 0.67. The statements which had the highest average nesting level were: the OPEN statement (2.18), the ELSE statement (1.75), the PERFORM statement (1.08), textual comments (1.07), and the GOTO statement (1.02). (See Appendix Figure 9.)

3.2. Added Sections Characteristics

The investigation of after which statement types sections of statements were added was done for the Procedure Division only, as the statement types present in the Data Division were severely limited. Sections of code were added only to programs in revisions 0 through 6 inclusive. The results of this investigation will be presented as follows:

- 1) add_sections[TYPE,LEVEL],
- 2) ANL_Add_Sections[TYPE], and
SD_ANL_Add_Sections[TYPE].

3.2.1. add_sections[TYPE,LEVEL]

3.2.1.1. Entire Program Collection

495 sections of statements were added in the Procedure Divisions of programs in the development phase. The statements after which sections of code were added had nesting levels of 0 through 2 inclusive. 52.93% of the sections were added at a nesting level of 0; 36.97% of the sections were added at a nesting level of 1; and 10.10% of the sections were added at a nesting level of 2. (See Appendix Figure 10.)

At level 0, sections of code were added most frequently after assignment statements and conditional statements. Sections of code were added most frequently after assignment statements at level 1. At level 2, sections of code were

added most frequently after branching statements and assignment statements. The places where sections of code were added most often at level 0 were: after MOVE statements (85), after IF statements (72), and after comments used for spacing purposes (42). At level 1, sections of code were added only after MOVE statements (92), GOTO statements (41), ELSE statements (30), PERFORM statements (13), and NEXT statements (7). Sections of code were added only after ELSE statements (24), NEXT statements (24), and GOTO statements (2) at level 2. (See Appendix Figure 10.)

3.2.1.2. Programs in the Early Phases of Development

440 sections of statements were added in the Procedure Divisions of programs in the later phases of development. The statements after which sections of code were added had nesting levels of 0 through 2 inclusive. 50.91% of the sections were added at a nesting level of 0; 38.18% of the sections were added at a nesting level of 1; and 10.91% of the sections were added at a nesting level of 2. (See Appendix Figure 11.)

At level 0, sections of code were added most frequently after assignment statements and conditional statements. Sections of code were added most frequently after assignment statements at level 1. At level 2, sections of code were added most frequently after branching statements and conditional statements. The places where sections of code were

added most often at level 0 were: after MOVE statements (74), after IF statements (72), and after comments used for spacing purposes (27). At level 1, sections of were added only after MOVE statements (89), GOTO statements (36), ELSE statements (26), PERFORM statements (12), and NEXT statements (5). Sections of code were added only after ELSE statements (24), and NEXT statements (24) at level 2. (See Appendix Figure 11.)

3.2.1.3. Programs in the Later Phases of Development

31 sections of statements were added in the Procedure Divisions of programs in the early phases of development. The statements after which sections of code were added had nesting levels of 0 and 1 only. 83.87% of the sections were added at a nesting level of 0; 16.13% of the sections were added at a nesting level of 1. (See Appendix Figure 12.)

At level 0, sections of code were added most frequently after assignment statements and comments. Sections of code were only added after branching statements and conditional statements at level 1. The places where sections of code were added most often at level 0 were: after MOVE statements (8) and after comments used for spacing purposes (8). At level 1, sections of were added only after GOTO statements (3), ELSE statements (1), and PERFORM statements (1). (See Appendix Figure 12.)

3.2.2. ANL_Add_Sections[TYPE] and SD_ANL_Add_Sections[TYPE]

3.2.2.1. Entire Program Collection

The average nesting level of a statement after which a section of code was added was 0.57. The places of insertion which had the highest average nesting level were: after a NEXT statement (1.45), after an ELSE statement (1.44), after a GOTO statement (0.88), after a PERFORM statement (0.81), and after a MOVE statement (0.52). All other places of insertion had an average nesting level of 0. (See Appendix Figure 13.)

3.2.2.2. Programs in the Early Phases of Development

The average nesting level of a statement after which a section of code was added was 0.60. The places of insertion which had the highest average nesting level were: after a ELSE statement (1.48), after a NEXT statement (1.47), after a PERFORM statement (0.86), after a GOTO statement (0.84), and after a MOVE statement (0.54). All other places of insertion had an average nesting level of 0. (See Appendix Figure 13.)

SD_ANL_Add_Sections[ALL] was 0.263. Except for the EXEC CICS statement (0.377) and the MOVE statement (0.179) the measure SD_ANL_Add_Sections[TYPE] was relatively low for most places of insertion. (See Appendix Figure 14.)

3.2.2.3. Programs in the Later Phases of Development

The average nesting level of a statement after which a section of code was added was 0.16. The places of insertion which had the highest average nesting level were: after a GOTO statement (0.75), after a PERFORM statement (0.50), and after a ELSE statement (0.50). All other places of insertion had an average nesting level of 0. (See Appendix Figure 13.)

SD_ANL_Add_Sections[ALL] was 0.148. Except for the ELSE statement (0.167) and the PERFORM statement (0.167) the measure SD_ANL_Add_Sections[TYPE] was relatively low for most places of insertion. (See Appendix Figure 14.)

3.3. Added Statements Characteristics

Statements were added only to programs in revisions 0 through 6 inclusive. The results of the investigation of what types of statements were added to programs during development will be presented as follows:

- 1) Adds[TYPE], %Adds[TYPE], and SD_%Adds[TYPE],
- 2) adds[TYPE,LEVEL],
- 3) ANL_Adds[TYPE], and SD_ANL_Adds[TYPE].

3.3.1. Adds[TYPE], %Adds[TYPE], and SD_%Adds[TYPE]

3.3.1.1. Entire Program Collection

A total of 4301 statements were added to the program collection during the development phase. 27.76% of the added statements were conditional statements; 27.62% of the added statements were assignment statements; 27.06% of the added statements were branching statements; 5.53% of the added statements were other statements; 4.93% of the added statements were declarations; 4.21% of the added statements were comments; 2.72% of the added statements were labels; and 0.16% of the added statements were input/output statements. The statement most often added to a program was the MOVE statement (1182). The GOTO statement (647), IF statement (622), and the ELSE statement (546) were also added frequently. (See Appendix Figure 15.)

3.3.1.2. Programs in the Early Phases of Development

A total of 3571 statements were added to programs in the early phases of development. 30.31% of the added statements were conditional statements; 29.17% of the statements were branching statements; 27.49% of the statements were assignment statements; 5.23% of the added statements were other statements; 3.49% of the added statements were declarations; 2.24% of the added statements were labels; 2.03% of the added statements were comments; and 0.08% of the added statements were input/output statements. The statement most often added to a program in the early phases of development was the MOVE statement (1031). The GOTO statement (603), IF statement (578), and the ELSE statement

(535) were also added frequently. (See Appendix Figure 16.)

The measure `SD_%Adds[TYPE]` was relatively low for most types of statements being added. The exceptions were: the EXAMINE statement (35.60), the EXEC CICS statement (6.77), and textual comments (3.78). (See Appendix Figure 18.)

3.3.1.3. Programs in the Later Phases of Development

A total of 260 statements were added to programs in the later phases of development. 31.92% of the added statements were assignment statements; 15.00% of the added statements were declarations; 12.69% of the added statements were branching statements; 10.77% of the added statements were comments; 8.08% of the added statements were labels; 8.08% of the added statements were other statements; and 1.93% of the added statements were input/output statements. The statement added most often to a program in the later phases of development was the MOVE statement (81). Declaration statements (29), IF statements (24), comments for spacing purposes (23), and GOTO statements (20) were also added frequently. (See Appendix Figure 17.)

The measure `SD_%Adds[TYPE]` was relatively low for most types of statements being added. The exceptions were: the MOVE statement (12.97), the IF statement (6.93), the EXEC CICS statement (5.02), and the NEXT statement (1.33). (See Appendix Figure 18.)

3.3.2. adds[TYPE,LEVEL]

3.3.2.1. Entire Program Collection

4022 statements were added in the Procedure Division of programs in the development phase. The statements added had nesting levels of 0 through 2 inclusive. 32.99% of the added statements had a nesting level of 0; 38.34 % of the added statements had a nesting level of 1; and 28.67% of the added statements had a nesting level of 2.

At level 0, assignment statements and branching statements were added most frequently, however, a wide variety of statements were added at level 0. MOVE statements (432), IF statements (399), and GOTO statements (71) were the most frequently added statements at level 0. At level 1, assignment statements were the most frequently added statements. 11 different statement types were added at level 1, with MOVE statements (708), IF statements (223), PERFORM statements (175) and GOTO statements (131) being the most frequently added statements. At level 2, only 4 different statement types were added: GOTO statements (445), ELSE statements (444), NEXT statements (222), and MOVE statements (42). Branching statements were the most frequently added statement type at level 2. (See Appendix Figure 19.)

3.3.2.2. Programs in the Early Phases of Development

3574 statements were added in the Procedure Divisions

of programs in the early phases of development. The statements added had nesting levels of 0 through 2 inclusive. 29.02% of the added statements had a nesting level of 0; 38.75% of the added statements had a nesting level of 1; and 32.23% of the added statements had a nesting level of 2.

At level 0, conditional statements and assignment statements were added most frequently. At level 1, assignment statements were added most frequently, while at level 2, branching statements were added most frequently. At level 0, 15 different types of statements were added with the most frequently added statements being the IF statement (356) and the MOVE statement (338). At level 1, 8 different types of statements were added, with the most frequently added statements being the MOVE statement (651) and the IF statement (222). At level 2, only 4 different types of statements were added. These statements were the ELSE statement (444), the GOTO statement (444), the NEXT statement (222), and the MOVE statement (42). (See Appendix Figure 20.)

3.3.2.3. Programs in the Later Phases of Development

221 statements were added in the Procedure Divisions of programs in the later phases of development. The statements added had nesting levels of 0 through 2 inclusive. 74.66% of the added statements had a nesting level of 0; 24.89% of the added statements had a nesting level of 1; and 0.45% of

the added statements had a nesting level of 2.

At level 0, assignment statements were added most frequently. At level 1, branching statements and assignment statements were added most frequently, while at level 2, only branching statements were added. At level 0, 13 different types of statements were added with the most frequently added statements being the MOVE statement (61), IF statement (23) and labels (21). At level 1, 8 different types of statements were added with the most frequently added statements being the MOVE statement (20) and the GOTO statement (17). At level 2, only one PERFORM statement was added. (See Appendix Figure 21.)

3.3.3. ANL_Adds[TYPE] and SD_ANL_Adds[TYPE]

3.3.3.1. Entire Program Collection

The average nesting level of an added statement was 0.96. The added statements which had the highest average nesting level were the ELSE statement (1.81), the NEXT statement (1.63), the GOTO statement (1.58), the PERFORM statement (0.88), the MOVE statement (0.67), the ON statement (0.50), the IF statement (0.36), and the ADD statement (0.33). All other statements had an average nesting level of 0. (See Appendix Figure 22.)

3.3.3.2. Programs in the Early Phases of Development

The average nesting level of an added statement was 1.03. The added statements which had the highest average nesting level were the ELSE statement (1.83), the NEXT statement (1.65), the GOTO statement (1.63), the PERFORM statement (0.92), the MOVE statement (0.71), and the IF statement (0.38). All other statements had an average nesting level of 0. (See Appendix Figure 22.)

SD_ANL_Adds[ALL] was 0.512. The measure SD_ANL_Adds[TYPE] was relatively low for most types of statements being added. The exceptions were: the MOVE statement (0.630), the NEXT statement (0.350), the PERFORM statement (0.346), and the ELSE statement (0.337). (See Appendix Figure 23.)

3.3.3.3. Programs in the Later Phases of Development

The average nesting level of an added statement was 0.51. The added statements which had the highest average nesting level were the ELSE statement (1.00), the GOTO statement (0.95), the PERFORM statement (0.43), the MOVE statement (0.25), textual comments (0.20), and the IF statement (0.04). All other statements had an average nesting level of 0. (See Appendix Figure 22.)

SD_ANL_Adds[ALL] was 0.000. Except for the MOVE statement (0.230) and the SET statement (0.167) the measure SD_ANL_Adds[TYPE] was relatively low for most types of statements being added. (See Appendix Figure 23.)

3.4. Changed Statements

Statements were changed in programs of all revisions. The results of the investigation of what types of statements were changed in programs during development will be presented as follows:

- 1) Changes[TYPE], Changes %[TYPE], %Changes[TYPE], and SD_%Changes[TYPE],
- 2) changes[TYPE, LEVEL],
- 3) ANL_Changes[TYPE] and SD_ANL_Changes[TYPE],
- 4) CHANGES[TYPE1,TYPE2] and CHANGES %[TYPE1,TYPE2],
- 5) The nesting environment of changed IF/ELSE statements.

3.4.1. Changes[TYPE], Changes %[TYPE], %Changes[TYPE], and SD_%Changes[TYPE]

3.4.1.1. Entire Program Collection

A total of 3497 statements were changed in the program collection during the development phase. 63.45% of the changed statements were assignment statements; 12.50% of the changed statements were conditional statements; 10.75% of the changed statements were branching statements; 6.68% of the changed statements were other statements; 4.06% of the changed statements were declarations; 2.09% of the changed statements were comments; 0.46% of the changed statements

were labels; and 0.11% of the changed statements were input/output statements. The statement changed most often in a program was the MOVE statement (2209). IF statements (266), PERFORM statements (180), ELSE statements (167), and the GOTO statements (146) were also changed frequently. (See Appendix Figure 24.)

2.22% of all statements were changed. The statement which was most likely to be changed was the REWRITE statement. 33.33% of all REWRITE statements were changed. EXAMINE statements (22.88%), OPEN statements (5.88%), and ON statements (5.13%) were also changed frequently. (See Appendix Figure 24.)

3.4.1.2. Programs in the Early Phases of Development

A total of 2465 statements were changed in programs in the early phases of development. 66.33% of the changed statements were assignment statements; 10.63% of the changed statements were branching statements; 10.47% of the changed statements were conditional statements; 7.91% of the changed statements were other statements; 3.65% of the changed statements were declarations; 0.85% of the changed statements were comments; 0.08% of the changed statements were labels; and 0.08% of the changed statements were input/output statements. The statements most often changed in a program were: MOVE statements (1633), PERFORM statements (144), IF statements (138), EXAMINE statements (134),

and ELSE statements (118). (See Appendix Figure 25.)

2.45% of all statements in early revision programs were changed. The statement which is most likely to be changed is the REWRITE statement. 33.33% of all REWRITE statements were changed. EXAMINE statements (23.34%) and DISPLAY statements (10.00%) were also changed frequently. (See Appendix Figure 25.)

The measure SD_%Changes[TYPE] was relatively low for most statement types. The exceptions were: the EXAMINE statement (35.60), the ELSE statement (15.40), the CALL statement (9.93), the NEXT statement (7.95), the IF statement (7.25), the EXEC CICS statement (6.77), and comments for spacing purposes (4.53). (See Appendix Figure 27.)

3.4.1.3. Programs in the Later Phases of Development

A total of 288 statements were changed in programs in the later phases of development. 69.10% of the changed statements were assignment statements; 11.11% of the changed statements were branching statements; 10.07% of the changed statements were conditional statements; 3.82% of the changed statements were declarations; 3.47% of the changed statements were other statements; 1.04% of the changed statements were labels; 0.69% of the changed statements were comments; and 0.59% of the changed statements were input/output statements. The MOVE statement (198), the IF statement (19), and the GOTO statement (16) were changed most frequently. (See

Appendix Figure 26.)

0.83% of all statements in programs in the later phases of development were changed. The statement which was most likely to be changed was the REWRITE statement. 100.00% of the REWRITE statements were changed. DELETE statements (25.00%), SET statements (20.00%), and NEXT statements (11.11%) were also changed frequently. (See Appendix Figure 26.)

The measure `SD_Changes[TYPE]` was relatively low for most statement types. The exceptions were: the PERFORM statement (5.20), the GOTO statement (5.10), the EXEC CICS statement (5.02), and labels (1.20). (See Appendix Figure 27.)

3.4.2. `changes[TYPE,LEVEL]`

3.4.2.1. Entire Program Collection

3344 statements were changed in the Procedure Divisions of programs in the development phase. The statements changed had nesting levels of 0 through 5 inclusive. 21.38% of the changed statements were at level 0; 63.61% of the changed statements were at level 1; 12.92% of the changed statements were at level 2; 1.56% of the changed statements were at level 3; 0.09% of the changed statements were at level 4; and 0.45% of the changed statements were at level 5. (See Appendix Figures 28a and 28b.)

Assignment statements were most frequently changed at levels 0,1, and 3. At level 2, assignment statements, conditional statements, and branching statements were changed most frequently. At level 4, conditional statements and branching statements were the only statements changed. At level 5, only branching statements, conditional statements, and assignment statements were changed. The statements most often changed at level 0 were: MOVE statements (377), and IF statements (144). At level 1, the statements changed most often were: MOVE statements (1585), PERFORM statements (146), and EXAMINE statements (128). The statements most often changed at level 2 were: MOVE statements (197) and IF statements (125). At level 3, the statement changed most frequently was the MOVE statement (48). The only statements which changed at level 4 were: an IF statement (1), a PERFORM statement (1), and a GOTO statement (1). At level 5, the only statements which were changed were: the ELSE statement (5), the GOTO statement (3), the NEXT statement (3), the MOVE statement (2), and the PERFORM statement (2). (See Appendix Figures 28a and 28b.)

3.4.2.2. Programs in the Early Phases of Development

2368 statements were changed in the Procedure Divisions of programs in the early phases of development. Only statements at levels 0 through 3 inclusive were changed. 16.93% of the changed statements were at level 0; 69.05% of the changed statements were at level 1; 12.63% of the changed

statements were at level 2; while 1.39% of the changed statements were at level 3. (See Appendix Figure 29.)

Assignment statements were most frequently changed at all levels. At level 0, the most frequently changed statements were: MOVE statements (239), IF statements (47), EXEC CICS statements (43), and GOTO statements (29). The most frequently changed statements at level 1 were: MOVE statements (1232), EXAMINE statements (128), and PERFORM statements (126). At level 2, the most frequently changed statements were: MOVE statements (131) and ELSE statements (92). Only MOVE statements (32) and PERFORM statements (1) were changed at level 3. (See Appendix Figure 29.)

3.4.2.3. Programs in the Later Phases of Development

275 statements were changed in the Procedure Divisions of programs in the later phases of development. Statements were changed at levels 0 through 5 inclusive. 28.00% of the changed statements were at level 0, 59.64% of the changed statements were at level 1; 4.73% of the changed statements were at level 2; 1.09% of the changed statements were at level 3; 1.09% of the changed statements were at level 4; and 5.45% of the statements were at level 5. (See Appendix Figures 30a and 30b.)

Assignment statements and conditional statements were most frequently changed at levels 0 and 1. At levels 2 and 5, only assignment statements, conditional statements, and

branching statements were changed. At levels 3 and 4, only conditional statements and branching statements were changed. The most frequently changed statements at level 0 were: MOVE statements (41) and IF statements (14). The MOVE statement (149) was the most frequently changed statement at level 1. At level 2, the only statements which changed were: the MOVE statement (6), the GOTO statement (4), the ELSE statement (2), and the IF statement (1). The only statements which were changed at levels 3 and 4 were: an IF statement (1), a PERFORM statement (1), and a GOTO statement (1). At level 5, the only statements which changed were: the ELSE statement (5), the GOTO statement (3), the NEXT statement (3), the MOVE statement (2), and the PERFORM statement (3). (See Appendix Figures 30a and 30b.)

3.4.3. ANL_Changes[TYPE] and SD_ANL_Changes[TYPE]

3.4.3.1. Entire Program Collection

The average nesting level of a changed statement was 0.97. The changed statements which had the highest average nesting level were the ELSE statement (1.86), the NEXT statement (1.68), the GOTO statement (1.22), the PERFORM statement (1.11), the ON statement (1.00), the OPEN statement (1.00), the MOVE statement (0.97), the EXAMINE statement (0.95), the ADD statement (0.90), textual comments (0.75), the IF statement (0.57), and the EXEC CICS statement (0.13). All other statements had an average nesting level

of 0. (See Appendix Figure 31.)

3.4.3.2. Programs in the Early Phases of Development

The average nesting level of a changed statement was 0.98. The changed statements which had the highest average nesting level were the ELSE statement (1.77), the ADD statement (1.50), the NEXT statement (1.37), the GOTO statement (1.12), the PERFORM statement (1.05), the OPEN statement (1.00), the EXAMINE statement (0.96), the MOVE statement (0.95), the IF statement (0.78), and the EXEC CICS statement (0.19). All other statements had an average nesting level of 0. (See Appendix Figure 31.)

SD_ANL_Changes[ALL] was 0.000. The measure SD_ANL_Changes[TYPE] was relatively low for most statement types. The exceptions were: the MOVE statement (0.468), the IF statement (0.468), and the OPEN statement (0.232). (See Appendix Figure 32.)

3.4.3.3. Programs in the Later Phases of Development

The average nesting level of an changed statement was 1.04. The changed statements which had the highest average nesting level were the NEXT statement (4.00), the ELSE statement (3.20), the GOTO statement (2.19), the MOVE statement (0.86), the IF statement (0.58), and the EXEC CICS statement (0.11). All other statements had an average nesting level of 0. (See Appendix Figure 31.)

SD_ANL_Changes[ALL] was 0.000. The measure SD_ANL_Changes[TYPE] was relatively low for most statement types. The exceptions were: the PERFORM statement (0.425), the MOVE statement (0.338), the IF statement (0.134). (See Appendix Figure 32.)

3.4.4. CHANGES[TYPE1,TYPE2] and CHANGES%[TYPE1,TYPE2]

The measures CHANGES[TYPE1,TYPE2] and CHANGES%[TYPE1,TYPE2] were considered only for the program collection as a whole. In general, the changes occurring can be classified into the following categories:

- 1) changes in which TYPE1 and TYPE2 were the same,
- 2) changes where the resulting type, TYPE2, was a comment,
- 3) changes which resulted in a higher level of nesting than the original statement, and
- 4) changes in which TYPE1 and TYPE2 statements perform similar functions.

DELETE statements, OPEN statements, and REWRITE statements had category one changes only. Declarations, VALUE clauses, ADD statements, ELSE statements, CALL statements, labels, and SET statements had both category one changes and category two changes. MOVE statements and EXEC CICS statements had categories 1, 2, and 3 changes. GOTO statements, PERFORM statements, and EXAMINE statements had categories 1,

2, 3, and 4 changes. NEXT statements had categories 1, 2, and 4 changes. IF statements had categories 1 and 2 changes, as well as having 3 IF statements change to a MOVE statement. Comments had category 1 changes, as well as having 48 comments change to a Procedure Division statement. (See Appendix Figure 33.)

3.4.5. The Nesting Environment of Changed IF/ELSE Statements

The nesting environment of changed IF statements and ELSE statements was considered only for the program collection as a whole.

The deepest level of nesting of IF statements was level 7. IF statements which were nested internally to a depth of level 6 or 7 had no changes in their encompassing scope. A total of 27 IF statements were nested to a depth of level 5. In the encompassing scope of these statements six IF statements (22.22%) were changed at level 0; 1 IF statement (3.70%) was changed at levels 1, 2, 3, and 4; one ELSE statement (5.26%) was changed at level 1; 15 ELSE statements (7.89%) were changed at level 2; and 5 ELSE statements (26.32%) were changed at level 5. 102 IF statements were nested to a depth of level 4. In the encompassing scope of these statements, 6 IF statements (5.88%) were changed at level 0; 1 IF statement (0.98%) was changed at level 2; and 37 ELSE statements (66.07%) were changed at level 2. A

total of 393 IF statements were nested to a depth of level 3. In the encompassing scope of these statements, 13 IF statements (3.31%) were changed at level 0; 39 IF statements (9.92%) were changed at level 1; 23 IF statements (5.85%) were changed at level 2; and 59 ELSE statements (26.82%) were changed at level 2. 2412 IF statements were nested to a depth of level 2. In the encompassing scope of these statements, 17 IF statements (0.70%) were changed at level 0; 55 IF statements (2.28%) were changed at level 1; 3 ELSE statements (0.10%) were changed at level 1; and 14 ELSE statements (0.46%) were changed at level 2. A total of 8764 IF statements were nested to a depth of level 1. In the encompassing scope of these statements, 102 IF statements (1.16%) were changed at level 0; and 32 ELSE statements (1.82%) were changed at level 1. Due to a syntax error in one of the COBOL programs, one deleted ELSE statement was classified as being at level 0. (See Appendix Figures 43-45, 48-49.)

3.5. Deleted Statements Characteristics

Statements were deleted only from programs in revisions 0 through 6 inclusive. The results of the investigation of what types of statements were deleted from programs during development will be presented as follows:

- 1) Deletes[TYPE], Deletes%(TYPE), %Deletes[TYPE], and
SD_%Deletes[TYPE],
- 2) deletes[TYPE,LEVEL],

- 3) ANL_Deletes[TYPE] and SD_ANL_Deletes[TYPE],
- 4) The nesting environment of deleted IF/ELSE statements.

3.5.1. Deletes[TYPE], Deletes%(TYPE), %Deletes[TYPE], and SD_%Deletes[TYPE]

3.5.1.1. Entire Program Collection

A total of 2022 statements were deleted from the program collection during the development phase. 32.69% of the deleted statements were declarations; 27.94% of the deleted statements were assignment statements; 16.32% of the deleted statements were comments; 12.96% of the deleted statements were conditional statements; 7.37% of the deleted statements were branching statements; 1.68% of the deleted statements were labels; 0.94% of the deleted statements were other statements; and 0.10% of the deleted statements were input/output statements. The statement most often deleted from a program was the MOVE statement (565). Declaration statements (545), textual comments (227), IF statements (210), and the spacing comments (103) were also frequently deleted. (See Appendix Figure 34.)

1.28% of all statements were deleted. The statement which was most likely to be deleted was the DISPLAY statement. 14.29% of all DISPLAY statements were deleted. OPEN statements (5.88%), textual comments (3.97%), and REDEFINES

statements (3.81) were also deleted frequently. (See Appendix Figure 34.)

3.5.1.2. Programs in the Early Phases of Development

A total of 1047 statements were deleted from programs in the early phases of development. 28.46% of the deleted statements were declaration statements; 28.18% of the deleted statements were comments; 25.50% of the deleted statements were assignment statements; 11.27% of the deleted statements were conditional statements; 4.20% of the deleted statements were branching statements; 1.15% of the deleted statements were labels; 1.15% of the deleted statements were other statements; and 0.10% of the deleted statements were input/output statements. The statements deleted most often were: MOVE statements (267), declaration statements (235), textual comments (208), IF statements (95), and comments for spacing purposes (87). (See Appendix Figure 35.)

1.04% of all statements in early revision programs were deleted. The statement which was most likely to be deleted was the DISPLAY statement. 20.00% of all DISPLAY statements were deleted. Useful comments (6.97%) were also deleted frequently. (See Appendix Figure 35.)

The measure `SD_%Deletes[TYPE]` was relatively low for most statement types. The exceptions were: the REDEFINES statement (11.71), the MOVE statement (8.69), the EXEC CICS

statement (6.99), and the GOTO statement (1.32). (See Appendix Figure 37.)

3.5.1.3. Programs in the Later Phases of Development

A total of 170 statements were deleted from programs in the later phases of development. 67.75% of the deleted statements were assignment statements; 12.94% of the deleted statements were conditional statements; 9.41% of the deleted statements were declarations; 4.71% of the deleted statements were comments; 2.35% of the deleted statements were branching statements; 1.76% of the deleted statements were other statements; 0.59% of the deleted statements were input/output statements; and 0.59% of the deleted statements were labels. The MOVE statement (115) and the IF statement (21) were deleted most frequently. (See Appendix Figure 36.)

0.49% of all statements in programs in the later phases of development were deleted. The statement which was most likely to be deleted was the OPEN statement. 100.00% of all OPEN statements were deleted. Declaration Sections (45.45%) were also deleted frequently. (See Appendix Figure 36.)

The measure `SD_Deletes[TYPE]` was relatively low for most statement types. The exceptions were: the EXEC CICS statement (3.06), comments used for spacing purposes (1.42), and VALUE clauses (1.30). (See Appendix Figure 37.)

3.5.2. deletes[TYPE, LEVEL]

3.5.2.1. Entire Program Collection

1034 statements were deleted from the Procedure Divisions of programs in the development phase. The statements deleted had nesting levels of 0 through 5 inclusive. 46.78% of the deleted statements were from level 0; 44.40% of the deleted statements were from level 1; 7.44% of the deleted statements were from level 2; 0.23% of the deleted statements were from level 3; 0.23% of the deleted statements were from level 4; and 0.92% of the deleted statements were from level 5. (See Appendix Figures 38a and 38b.)

Comments and conditional statements were most frequently deleted at level 0. Assignment statements were most frequently deleted at level 1. At level 2, branching statements, conditional statements, and assignment statements were deleted most frequently. At levels 3, 4, and 5, only branching statements and conditional statements were deleted. The statements most often deleted at level 0 were: textual comments (207), IF statements (190), and MOVE statements (80). At level 1, the most often deleted statements were: MOVE statements (464), and GOTO statements (47). The statements most often deleted at level 2 were: ELSE statements (32), MOVE statements (21), GOTO statements (18), and NEXT statements (16). At levels 3 and 4, the only statements deleted were: an IF statement (1), a PERFORM statement

(1), and a GOTO statement (1). At level 5, the only statements which were deleted were: the ELSE statement (5), the GOTO statement (3), the NEXT statement (3), and the PERFORM statement (1). (See Appendix Figures 38a and 38b.)

3.5.2.2. Programs in the Early Phases of Development

715 statements were deleted from the Procedure Divisions of programs in the early phases of development. Only statements at levels 0 through 2 inclusive were deleted. 59.30% of the deleted statements were from level 0; 34.83% of the deleted statements were from level 1; and 5.87% of the deleted statements were from level 2. (See Appendix Figure 39.)

Comments were the most frequently deleted statement type at level 0. Assignment statements were most frequently deleted at levels 1 and 2. At level 0, the most frequently deleted statements were: textual comments (205), IF statements (88), comments for spacing purposes (56), and MOVE statements (48). The most frequently deleted statements at level 1 were: MOVE statements (198) and GOTO statements (23). At level 2, the only statements deleted were: the MOVE statement (21), the IF statement (14), and the NEXT statement (7). (See Appendix Figure 39.)

3.5.2.3. Programs in the Later Phases of Development

154 statements were deleted from the Procedure

Divisions of programs in the later phases of development. Only level 0 and level 1 statements were deleted. 28.57% of the deleted statements were from level 0, and 71.43% of the deleted statements were from level 1. (See Appendix Figure 40.)

Conditional statements and assignment statements were most frequently deleted from level 0, while assignment statements were most frequently deleted from level 1. Seven different statement types were deleted from level 0 with the IF statement (21), the MOVE statement (12), and comments for spacing purposes (8) being the most frequently deleted. Five different statement types were deleted from level 1, with the MOVE statement (104) being the most frequently deleted. (See Appendix Figure 40.)

3.5.3. ANL_Deletes[TYPE] and SD_ANL_Deletes[TYPE]

3.5.3.1. Entire Program Collection

The average nesting level of a deleted statement was 0.65. The deleted statements which had the highest average nesting level were the ELSE statement (2.04), the NEXT statement (1.90), the PERFORM statement (1.44), the GOTO statement (1.31), the DISPLAY statement (1.00), the OPEN statement (1.00), the MOVE statement (0.90), the EXEC CICS statement (0.13), and the IF statement (0.12). All other statements had an average nesting level of 0. (See Appendix

Figure 41.)

3.5.3.2. Programs in the Early Phases of Development

The average nesting level of a deleted statement was 0.47. The deleted statements which had the highest average nesting level were the ELSE statement (1.61), the NEXT statement (1.44), the DISPLAY statement (1.00), the GOTO statement (0.96), the MOVE statement (0.90), the PERFORM statement (0.66), and the IF statement (0.07). All other statements had an average nesting level of 0. (See Appendix Figure 41.)

SD_ANL_Deletes[ALL] was 0.305. The measure SD_ANL_Deletes[TYPE] was relatively low for most statement types. The exceptions were: the NEXT statement (0.648), the ELSE statement (0.648), the MOVE statement (0.543), the GOTO statement (0.416), the EXEC CICS statement (0.377), and the DISPLAY statement (0.167). (See Appendix Figure 42.)

3.5.3.3. Programs in the Later Phases of Development

The average nesting level of a deleted statement was 0.71. The deleted statements which had the highest average nesting level were the ELSE statement (1.00), the OPEN statement (1.00), the MOVE statement (0.90), the GOTO statement (0.66), and the EXEC CICS statement (0.66). All other statements had an average nesting level of 0. The exceptions were: the MOVE statement (0.290), the GOTO statement

(0.232), the ELSE statement (0.167), and the OPEN statement (0.167). (See Appendix Figure 41.)

SD_ANL_Deletes[ALL] was 0.000. The measure SD_ANL_Deletes[TYPE] was relatively low for most statement types. The exceptions were: the MOVE statement (0.290), the GOTO statement (0.232), the ELSE statement (0.167), and the OPEN statement (0.167). (See Appendix Figure 42.)

3.5.4. The nesting environment of deleted IF/ELSE Statements

The nesting environment of deleted IF statements and ELSE statements was considered only for the program collection as a whole.

The deepest level of nesting of IF statements was level 7. IF statements which were nested internally to a depth of level 6 or 7 had no deletions in their encompassing scope. A total of 27 IF statements were nested to a depth of level 5. In the encompassing scope of these statements ten IF statements (37.03%) were deleted at level 0; 1 IF statement (3.70%) was deleted at levels 1, 2, 3, and 4; two ELSE statements (10.53%) were deleted at level 2; and 5 ELSE statements (26.32%) were deleted at level 5. 102 IF statements were nested to a depth of level 4. In the encompassing scope of these statements, 16 IF statements (15.69%) were deleted at level 0. A total of 393 IF statements were nested to a depth of level 3. In the encompassing scope of

these statements, 17 IF statements (4.33%) were deleted at level 0; 3 IF statements (0.76%) were deleted at level 1; and 13 ELSE statements (59.09%) were deleted at level 2. 2412 IF statements were nested to a depth of level 2. In the encompassing scope of these statements, 19 IF statements (0.79%) were deleted at level 0; 13 IF statements (0.54%) were deleted at level 1; 13 ELSE statements (0.43%) were deleted at level 1; and 17 ELSE statements (0.56%) were deleted at level 2. A total of 8764 IF statements were nested to a depth of level 1. In the encompassing scope of these statements, 128 IF statements (1.46%) were deleted at level 0. (See Appendix Figures 43,46,47,50,51.)

Chapter 4

Conclusions and Suggestions for Future Work

Chapter three presented the results of this study in an objective manner. The code was shown to be unstructured as well as poorly documented. The corporation from which the programs used for this analysis were obtained does not require that code be commented, thus explaining the lack of documentation. However, the corporation does encourage the use of structured programming, yet programmers are resisting the management's directive to produce structured code.

The COBOL programs analyzed were shown to have a relatively low average nesting level (0.67). Statements which were nested internally to a depth of levels 6 or 7 did not exhibit any modifications, and furthermore, IF and ELSE statements in the encompassing scope of the nested IF statements were not modified. This suggests that the logic of these deeply nested statements was initially well defined. However, it was shown for all other depths of nesting that modifications do occur quite frequently within the encompassing scope of the nested IF statement. As shown in Figures 48-52, the actual percentage of IF/ELSE statements which were modified decreased significantly as the internal nesting level of the encompassing IF statement decreased. Considering that many other statements were changed/deleted at each level and that further analysis would most likely

show that statement types other than the IF and ELSE statements were also being modified within the encompassing scope of nested IF statements, the intuitive belief that highly nested code will result in additional complexity as well as modifications to the program was shown to have some foundation.

The resulting types of statements which changed were not all that surprising. The majority of the statements being changed were changed to a statement of the same type as the original statement or were commented out for debugging purposes. In addition, many statements were simply changed by adding a level of nesting to the original statement (ie. The statement MOVE X TO Y. changing to IF X = 2 MOVE X TO Y.). Disappointingly, 10.56% of the changed PERFORM statements were changed to GOTO statements, while only 2.06% of the changed GOTO statements changed to PERFORM statements. This again reflects the reluctance of the programmers to produce structured code.

As expected, programs in the early phases of revision were modified much more frequently than programs in the later phases of revision. Additions and deletions occurred only in revisions 0 through 6 of the programs. However, changes occurred in all revisions. In revisions 7 through 12 inclusive, the only modifications which were occurring were the changing of existing statements. The types of modifications which occurred throughout the development pro-

cess, suggest that the activity which occurred in the later phases of program development was restricted to fine tuning of the code (ie. altering existing statements).

The number of added sections and the nesting level of the place of insertion of code both decreased as the number of revisions of the code increased. The average nesting level of an added section in the early phases of development was 0.60 while the average nesting level of an added section in the later phase of development was 0.16. This could indicate that the general structure of sections of highly nested code were being fairly well defined initially as no major changes to the structure of nested statements at the deeper levels were occurring.

The number and nesting level of added statements also decreased as the code progressed from revision to revision. The average nesting level for statements added to programs in the early phases of revision was 1.03 while the average nesting level of statements added in later phases of revision was 0.51. In addition, a much larger percentage of branching and conditional statements were being added in programs in the earlier phases of development than in programs in the later phases of development. 30.31% of the added statements in the early phases of revision were conditional statements while 11.54% of the added statements in the later phases of revision were conditional statements. 29.17% of the added statements in the early phases of revision

sion were branching statements while 12.69% of the added statements in the later phases of revision were branching statements. The lower percentage of conditional and branching statements being added at later phases of development indicates that additions to the conditional and branching logic decrease as the program development progresses.

It should also be noted that a rather large amount of declarations were added to some revision 6 programs. Previous research [LANC86] showed that many declaration modifications are indicative of design deficiencies. The programs which exhibited large amounts of declarative additions were found to have had hazy specifications and target users who were indecisive about their needs. Therefore, it was necessary to modify the data structure late in the development process.

The average nesting level of changed statements in programs in the later phases of development was actually slightly higher than the average nesting level of changed statements in the earlier phases of development (1.04 and 0.98). The standard deviation for the average nesting level of a changed statement in the later revision programs (0.424) and for the earlier phases (0.000) enforce the accuracy of the first measure. Further investigation showed that the average nesting level of a changed branching statement and conditional statement actually increased with revision number while the average nesting level of all other

change statement types decreased with revision number. This indicates that fine tuning of the more deeply nested conditional and branching statements was occurring in the later phases of the development process.

The average nesting level of statements deleted from programs in the later phases of development (0.71) was slightly higher than the average nesting level of statements deleted from programs in the earlier phases of development (0.47). The standard deviation for the average nesting level of a deleted statement in the later phased of development was (0.00), while the standard deviation for the average nesting level of a deleted statement in the early phases of development was (0.305). The relatively high value for the standard deviation for the average nesting level of a deleted statement in the later revision programs (0.305) discounts the measure. Further investigation showed that the average nesting level of most deleted statements types in programs in the later phases of development was equivalent or slightly less than for the programs in the earlier phases of development. The exception being the EXEC CICS statement.

The largest percentage of statements being deleted during the later phases of development were assignment statements. However, during the early phases of development a large percentage of declarations, comments (mainly statements which have been commented out), and assignment state-

ments were deleted.

The general characteristics of a program in the early phases of development were:

- 1) a large number and a wide variety of statements being added, changed, and deleted,
- 2) the average nesting level of the place of insertion of a section of code being 0.60 (standard deviation of 0.263),
- 3) the average nesting level of an added statement being 1.03 (standard deviation of 0.512),
- 4) all changes occurring at levels 0 through 2 inclusive,
- 5) the average nesting level of a change being 0.98 (standard deviation of 0.000),
- 6) the average nesting level of a deletion being 0.47 (standard deviation of 0.47).

The general characteristics of a program in the later phases of development were:

- 1) few additions and deletions of statements (no additions or deletions occurring past revision 6),
- 2) few changed statements,
- 3) the average nesting level of the place of

insertion of a section of code being 0.16 (standard deviation of 0.148),

- 4) the average nesting level of an added statement being 0.51 (standard deviation of 0.000),
- 5) smaller percentages of conditional and branching statements being added,
- 6) the average nesting level of a change being 1.04 (standard deviation of 0.424),
- 7) changed branching and conditional statements being at deeper levels of nesting than with the earlier phases of revision,
- 8) the average nesting level of a deletion being 0.71 (standard deviation of 0.000),
- 9) the majority of deleted statements being assignment statements.

The general characteristics discussed above suggest that the method used for program development was a quasi top down methodology having the following steps:

- 1) attempt to encode the entire program as a whole,
- 2) begin debugging the simplest cases first, making all necessary additions, deletions, and changes to remedy all errors encountered,

- 3) progress to debugging the more difficult cases
(more deeply nested statements).

The data used in this research was not gathered in an orderly fashion, and thus the amount of useful conclusions which could be drawn from the data collection was severely limited. Future work in the area of investigating software development should emphasize the controlled collection of "real world" data to allow detailed investigation of the progression of the software development in an orderly fashion. Given a complete program collection and an indepth knowledge of the program development environment, interesting correlations between the types of code transformations occurring and the revision number of the code should be possible and should also lead to definitive patterns in the code development process.

In addition, emphasis should be placed upon analyzing the environment of a modified statement. Even though this work looked only at the environment of modified IF statements and ELSE statements, it has shown that consideration of the environment of a modified statement is definitely needed in any software measure.

Lastly, a larger collection of programs of this nature should be analyzed to verify and extend this work.

References

- [ALBR83] Albrecht, A. J., and J. E. Gaffney, Jr. Software function, sources lines of code, and development effort prediction: a Software Science validation. IEEE Transactions on Software Engineering SE-9, 6 (November 1983): 639-647.
- [BAKE80] Baker, A. L., and S.H. Zweben. A comparison of measures of control flow complexity. IEEE Transactions on Software Engineering SE-6, 6 (November 1980): 39-45.
- [BASIB3] Basili, V.R., R. W. Selby, Jr., and T. Y. Phillips. Metric analysis and data validation across FORTRAN projects. IEEE Transactions on Software Engineering SE-9, 6 (November 1983): 652-663.
- [BASIB4] Basili, V. R., and B. T. Perricone. Software errors and complexity: an empirical investigation. Communications of the ACM 27, 1 (January 1984): 42-52.
- [BERN84] Berns G. M. Assessing Software Maintainability. Communications of the ACM 27, 1 (January 1984): 14-23.
- [CONT86] Conte, S. D., H. E. Dunsmore, and V. Y. Shen. Software Engineering Metrics and Models. Menlo Park, California: The Benjamin/Cummings Publishing Company, 1986.
- [DEMA82] DeMarco T. Controlling Software Projects: Management, Measurement, and Estimation. New York: Yourdon Press, 1982.
- [DIJK68] Dijkstra, E. W. Go to statements considered harmful. Communications of the ACM 11, 2 (March 1968): 147-148.
- [DUNH83] Dunham, J. R., and E. Kruesi. The measurement task area. IEEE Computer 16, 11 (November 1983): 47-54.
- [GUST85] Gustafson, D. A., A. C. Melton Jr., and C. S. Hsieh. An analysis of software changes during maintenance and enhancement. Department of Computer Science, Kansas State University, Manhattan, KS (1985).
- [HENR81] Henry, S., and D. Kafura. Software structure metrics based on information flow. IEEE Transactions on Software Engineering SE-7, 5 (September 1981): 510-518.

- [LANC86] Lanchbury, M. A. A model of successful patterns of progress during the integration of software. Master Thesis, Department of Computer Science, Kansas State University, Manhattan, KS (May 1986).
- [MCCA74] McCabe, T. J. A Complexity Measure. IEEE Transactions on Software Engineering SE-2, 4 (December 1976): 308-320.
- [MYER77] Myers, G. J. An extension to the cyclomatic measure of program complexity. SIGPLAN Notices 12, 10 (October 1977): 61-64.
- [PRES82] Pressman R. S. Software Engineering: A Practitioner's Approach. New York: McGraw-Hill, 1982.
- [SHEN83] Shen, V. Y., S. D. Conte, and H. E. Dunsmore. Software Science revisited: a critical analysis of the theory and its empirical support. IEEE Transactions on Software Engineering SE-9, 2 (March 1983): 155-165.
- [SHEN85] Shen, V. Y., T. J. Yu, S. M. Thebaut, and L. R. Paulsen. Identifying error-prone software - an empirical study. IEEE Transactions on Software Engineering SE-11, 4 (April 1985): 317-324.
- [VICK84] Vick C. R. and C. V. Ramamoorthy. Handbook of Software Engineering. New York: Van Nostrand Reinhold Company, 1984.
- [WALS77] Walston, C. E., and C. P. Felix. A method of programming measurement and estimation. IBM System Journal 16, 1 (1977): 54-73.
- [WOOD79] Woodward, M. R., M. A. Hennell, and D. Hedley. A measure of control flow complexity in program text. IEEE Transactions on Software Engineering SE-5, 1 (January 1979): 45-50.

Appendix

TYPE	Number[TYPE]	Number[TYPE] / Number[ALL]
ALL	157575	
comments	18551	11.77
spacing purposes	12839	8.15
textual	5712	3.62
declarations	36660	23.27
sections	113	0.07
declaration	30223	19.18
VALUE clauses	5721	3.63
redefines	603	0.38
assignments	53010	33.64
MOVE	51530	32.70
ADD	1404	0.89
SUBTRACT	54	0.03
COMPUTE	22	0.01
conditionals	21774	13.82
IF	15545	9.87
ELSE	5984	3.80
ON	78	0.05
AT END	167	0.11
branches	17223	10.93
CALL	428	0.27
PERFORM	5067	3.22
GOTO	9813	6.23
NEXT	2403	1.52
EXIT	7	0.00
input/output	474	0.30
DELETE	87	0.06
DISPLAY	7	0.00
OPEN	17	0.01
READ	66	0.04
REWRITE	6	0.00
WRITE	291	0.18
labels	6064	3.85
other statements	3819	2.42
EXAMINE	590	0.37
INSPECT	198	0.13
SEARCH	227	0.14
SET	240	0.15
EXEC CICS	2467	1.56
GOBACK	97	0.06

Figure 3. Number[TYPE] and Number[TYPE]/Number[ALL]
for all programs

TYPE	Number[TYPE]	Number[TYPE] / Number[ALL]
ALL	100469	
comments	8685	8.64
spacing purposes	5701	5.67
textual	2984	2.97
declarations	20568	20.47
sections	79	0.08
declaration	16460	16.38
VALUE clauses	3703	3.69
redefines	326	0.32
assignments	36602	36.43
MOVE	35500	35.33
ADD	1047	1.04
SUBTRACT	46	0.05
COMPUTE	9	0.01
conditionals	15165	15.09
IF	10520	10.47
ELSE	4450	4.43
ON	42	0.04
AT END	153	0.02
branches	12314	12.26
CALL	188	0.19
PERFORM	3101	3.09
GOTO	6836	6.80
NEXT	2186	2.18
EXIT	3	0.00
input/output	370	0.37
DELETE	72	0.07
DISPLAY	5	0.00
OPEN	10	0.01
READ	52	0.05
REWRITE	3	0.00
WRITE	228	0.23
labels	3988	3.97
other statements	3051	3.02
EXAMINE	574	5.71
INSPECT	198	0.20
SEARCH	200	0.20
SET	206	0.21
EXEC CICS	1806	1.80
GOBACK	67	0.07

Figure 4. Number[TYPE] and Number[TYPE]/Number[ALL]
for revision 0-2 programs

TYPE	Number[TYPE]	Number[TYPE] / Number[ALL]
ALL	34814	
comments	6118	17.57
spacing purposes	4492	12.90
textual	1626	4.67
declarations	8728	25.07
sections	11	0.03
declaration	7615	21.87
VALUE clauses	960	2.76
redefines	142	0.41
assignments	10548	30.30
MOVE	10358	39.75
ADD	176	0.51
SUBTRACT	2	0.01
COMPUTE	12	0.03
conditionals	4721	13.56
IF	3554	10.21
ELSE	1141	3.28
ON	18	0.05
AT END	8	0.02
branches	3101	8.91
CALL	187	0.54
PERFORM	1378	3.96
GOTO	1498	4.30
NEXT	36	0.10
EXIT	2	0.01
input/output	36	0.10
DELETE	4	0.01
DISPLAY	0	0.00
OPEN	1	0.00
READ	4	0.01
REWRITE	1	0.00
WRITE	26	0.07
labels	1270	3.65
other statements	292	0.84
EXAMINE	3	0.01
INSPECT	0	0.00
SEARCH	11	0.03
SET	11	0.03
EXEC CICS	252	0.72
GOBACK	15	0.04

Figure 5. Number[TYPE] and Number[TYPE]/Number[ALL]
for revision 6-11 programs

TYPE	Total	Level 0	Level 1	Level 2
ALL	115102	57686	42000	12567
comments	12738	11066	851	643
spacing purposes	10089	10089	0	0
textual	2649	977	851	643
assignments	53010	20526	27426	3740
MOVE	51530	20076	26470	3666
ADD	1404	423	907	74
SUBTRACT	54	27	27	0
COMPUTE	22	0	22	0
conditionals	21774	11938	5325	3753
IF	15545	11733	2969	557
ELSE	5984	3	2313	3196
ON	78	35	43	0
AT END	167	167	0	0
branches	17223	4756	7464	4418
CALL	428	424	4	0
PERFORM	5067	1451	2398	865
GOTO	9318	2657	4051	2415
NEXT	2403	217	1011	1138
EXIT	7	7	0	0
input/output	474	456	8	0
DELETE	87	87	0	0
DISPLAY	7	6	1	0
OPEN	17	0	7	0
READ	66	66	0	0
REWRITE	6	6	0	0
WRITE	291	291	0	0
labels	6064	6064	0	0
other statements	3819	2880	926	13
EXAMINE	590	139	450	1
INSPECT	198	0	198	0
SEARCH	227	227	0	0
SET	240	238	2	0
EXEC CICS	2467	2179	276	12
GOBACK	97	97	0	0

Figure 6a. Number[TYPE,LEVEL] for all programs (Levels 0-2)

TYPE	Level 3	Level 4	Level 5	Level 6	Level 7
ALL	1876	510	215	166	82
comments	128	7	9	9	25
spacing purposes	0	0	0	0	0
textual	128	7	9	9	25
assignments	939	243	36	68	32
MOVE	939	243	36	68	32
ADD	0	0	0	0	0
SUBTRACT	0	0	0	0	0
COMPUTE	0	0	0	0	0
conditionals	446	153	79	80	0
IF	164	62	35	25	0
ELSE	282	91	44	55	0
ON	0	0	0	0	0
AT END	0	0	0	0	0
branches	353	107	91	9	25
CALL	0	0	0	0	0
PERFORM	208	70	41	9	25
GOTO	135	37	23	0	0
NEXT	10	0	27	0	0
EXIT	0	0	0	0	0
input/output	10	0	0	0	0
DELETE	0	0	0	0	0
DISPLAY	0	0	0	0	0
OPEN	10	0	0	0	0
READ	0	0	0	0	0
REWRITE	0	0	0	0	0
WRITE	0	0	0	0	0
labels	0	0	0	0	0
other statements	0	0	0	0	0
EXAMINE	0	0	0	0	0
INSPECT	0	0	0	0	0
SEARCH	0	0	0	0	0
SET	0	0	0	0	0
EXEC CICS	0	0	0	0	0
GOBACK	0	0	0	0	0

Figure 6b. Number[TYPE,LEVEL] for all programs (Levels 3-7)

TYPE	Total	Level 0	Level 1	Level 2
ALL	77479	35568	31217	9335
comments	5989	5490	231	202
spacing purposes	4758	4758	0	0
textual	1231	732	231	202
assignments	36602	12024	21346	2494
MOVE	35500	11735	20565	2462
ADD	1047	256	759	32
SUBTRACT	46	33	13	0
COMPUTE	9	0	9	0
conditionals	15165	8298	3503	3045
IF	10520	8124	1975	307
ELSE	4450	3	1504	2738
ON	42	18	24	0
AT END	153	153	0	0
branches	12314	3204	5293	3587
CALL	188	184	4	0
PERFORM	3101	965	1663	339
GOTO	6836	1863	2759	2140
NEXT	2186	189	867	1108
EXIT	3	3	0	0
input/output	370	359	5	0
DELETE	72	72	0	0
DISPLAY	5	4	1	0
OPEN	10	0	4	0
READ	52	52	0	0
REWRITE	3	3	0	0
WRITE	228	228	0	0
labels	3988	3988	0	0
other statements	3051	2205	839	7
EXAMINE	574	124	449	1
INSPECT	198	0	198	0
SEARCH	200	200	0	0
SET	206	205	1	0
EXEC CICS	1806	1609	191	6
GOBACK	67	67	0	0

Figure 7a. Number[TYPE,LEVEL] for revision 0-2 programs
(Levels 0-2)

TYPE	Level 3	Level 4	Level 5	Level 6	Level 7
ALL	970	211	88	60	30
comments	48	3	3	3	9
spacing purposes	0	0	0	0	0
textual	48	3	3	3	9
assignments	569	113	18	26	12
MOVE	569	113	18	26	12
ADD	0	0	0	0	0
SUBTRACT	0	0	0	0	0
COMPUTE	0	0	0	0	0
conditionals	203	56	32	28	0
IF	69	23	13	9	0
ELSE	134	33	19	19	0
ON	0	0	0	0	0
AT END	0	0	0	0	0
branches	144	39	35	3	9
CALL	0	0	0	0	0
PERFORM	82	25	15	3	9
GOTO	52	14	8	0	0
NEXT	10	0	12	0	0
EXIT	0	0	0	0	0
input/output	6	0	0	0	0
DELETE	0	0	0	0	0
DISPLAY	0	0	0	0	0
OPEN	6	0	0	0	0
READ	0	0	0	0	0
REWRITE	0	0	0	0	0
WRITE	0	0	0	0	0
labels	0	0	0	0	0
other statements	0	0	0	0	0
EXAMINE	0	0	0	0	0
INSPECT	0	0	0	0	0
SEARCH	0	0	0	0	0
SET	0	0	0	0	0
EXEC CICS	0	0	0	0	0
GOBACK	0	0	0	0	0

Figure 7b. Number[TYPE,LEVEL] for revision 0-2 programs
(Levels 3-7)

TYPE	Total	Level 0	Level 1	Level 2
ALL	25361	15351	6154	2620
comments	5416	4345	534	416
spacing purposes	4190	4190	0	0
textual	1226	185	534	416
assignments	10548	6186	2852	1022
MOVE	10358	6073	2799	998
ADD	176	112	40	24
SUBTRACT	2	1	1	0
COMPUTE	12	0	12	0
conditionals	4721	2364	1441	529
IF	3554	2347	855	200
ELSE	1141	0	577	329
ON	18	9	9	0
AT END	8	8	0	0
branches	3101	897	1284	651
CALL	187	187	0	0
PERFORM	1378	250	493	460
GOTO	1498	450	766	191
NEXT	36	8	25	0
EXIT	2	2	0	0
input/output	36	36	0	0
DELETE	4	4	0	0
DISPLAY	0	0	0	0
OPEN	1	1	0	0
READ	4	4	0	0
REWRITE	1	1	0	0
WRITE	26	26	0	0
labels	1270	1270	0	0
other statements	292	254	38	0
EXAMINE	3	3	0	0
INSPECT	0	0	0	0
SEARCH	11	11	0	0
SET	11	11	0	0
EXEC CICS	252	214	38	0
GOBACK	15	15	0	0

Figure 8a. Number[TYPE,LEVEL] for revision 6-11 programs
(Levels 0-2)

TYPE	Level 3	Level 4	Level 5	Level 6	Level 7
ALL	746	271	71	104	44
comments	64	1	6	6	14
spacing purposes	0	0	0	0	0
textual	64	1	6	6	14
assignments	294	122	14	42	16
MOVE	294	122	14	42	16
ADD	0	0	0	0	0
SUBTRACT	0	0	0	0	0
COMPUTE	0	0	0	0	0
conditionals	223	89	25	50	0
IF	85	33	20	14	0
ELSE	138	56	5	36	0
ON	0	0	0	0	0
AT END	0	0	0	0	0
branches	165	58	26	6	14
CALL	0	0	0	0	0
PERFORM	96	39	20	6	14
GOTO	69	19	3	0	0
NEXT	0	0	3	0	0
EXIT	0	0	0	0	0
input/output	0	0	0	0	0
DELETE	0	0	0	0	0
DISPLAY	0	0	0	0	0
OPEN	0	0	0	0	0
READ	0	0	0	0	0
REWRITE	0	0	0	0	0
WRITE	0	0	0	0	0
labels	0	0	0	0	0
other statements	0	0	0	0	0
EXAMINE	0	0	0	0	0
INSPECT	0	0	0	0	0
SEARCH	0	0	0	0	0
SET	0	0	0	0	0
EXEC CICS	0	0	0	0	0
GOBACK	0	0	0	0	0

Figure 8b. Number[TYPE,LEVEL] for revision 6-11 programs
(Levels 3-7)

TYPE	all programs	revisions 0-2	revisions 6-11
ALL	0.67	0.71	0.63
comments	0.22	0.15	0.32
spacing purposes	0.00	0.00	0.00
textual	1.07	0.72	1.41
assignments	0.75	0.79	0.64
MOVE	0.75	0.79	0.64
ADD	0.75	0.79	0.48
SUBTRACT	0.50	0.36	0.50
COMPUTE	1.00	1.00	0.86
conditionals	0.72	0.71	0.84
IF	0.33	0.84	0.51
ELSE	1.75	1.74	1.84
ON	0.55	0.57	0.82
AT END	0.00	0.00	0.00
branches	1.07	1.07	1.15
PERFORM	1.08	0.92	1.52
GOTO	1.02	1.07	0.97
NEXT	1.44	1.45	1.11
EXIT	0.00	0.00	0.00
input/output	0.08	0.06	0.00
DELETE	0.00	0.00	0.00
DISPLAY	0.14	0.20	0.00
OPEN	2.18	2.20	0.00
READ	0.00	0.00	0.00
REWRITE	0.00	0.00	0.00
WRITE	0.00	0.00	0.00
labels	0.00	0.00	0.00
other statements	0.25	0.28	0.00
EXAMINE	0.77	0.78	0.00
INSPECT	1.00	1.00	0.00
SEARCH	0.00	0.00	0.00
SET	0.01	0.00	0.00
EXEC CICS	0.12	0.11	0.00
GOBACK	0.00	0.00	0.00

Figure 9. ANL[TYPE]

TYPE	Total	Level 0	Level 1	Level 2
ALL	495	262	183	50
comments	42	42	0	0
spacing purposes	42	42	0	0
textual	0	0	0	0
assignments	178	86	92	0
MOVE	177	85	92	0
ADD	1	1	0	0
SUBTRACT	0	0	0	0
COMPUTE	0	0	0	0
conditionals	126	72	30	24
IF	72	72	0	0
ELSE	54	0	30	24
ON	0	0	0	0
AT END	0	0	0	0
branches	107	20	61	26
CALL	2	2	0	0
PERFORM	16	3	13	0
GOTO	51	8	41	2
NEXT	38	7	7	24
EXIT	0	0	0	0
input/output	1	1	0	0
DELETE	0	0	0	0
DISPLAY	0	0	0	0
OPEN	0	0	0	0
READ	0	0	0	0
REWRITE	0	0	0	0
WRITE	1	1	0	0
labels	24	24	0	0
other statements	17	17	0	0
EXAMINE	0	0	0	0
INSPECT	0	0	0	0
SEARCH	2	2	0	0
SET	1	1	0	0
EXEC CICS	14	14	0	0
GOBACK	0	0	0	0

Figure 10. add_section[TYPE,LEVEL] for all programs

TYPE	Total	Level 0	Level 1	Level 2
ALL	440	224	168	48
comments	27	27	0	0
spacing purposes	27	27	0	0
textual	0	0	0	0
assignments	163	74	89	0
MOVE	163	74	89	0
ADD	0	0	0	0
SUBTRACT	0	0	0	0
COMPUTE	0	0	0	0
conditionals	122	72	26	24
IF	72	72	0	0
ELSE	50	0	26	24
ON	0	0	0	0
AT END	0	0	0	0
branches	94	17	53	24
CALL	1	1	0	0
PERFORM	14	2	12	0
GOTO	43	7	36	0
NEXT	36	7	5	24
EXIT	0	0	0	0
input/output	1	1	0	0
DELETE	0	0	0	0
DISPLAY	0	0	0	0
OPEN	0	0	0	0
READ	0	0	0	0
REWRITE	0	0	0	0
WRITE	1	1	0	0
labels	20	20	0	0
other statements	13	13	0	0
EXAMINE	0	0	0	0
INSPECT	0	0	0	0
SEARCH	1	1	0	0
SET	1	1	0	0
EXEC CICS	11	11	0	0
GOBACK	0	0	0	0

Figure 11. add_sections[TYPE,LEVEL] for revision 0-2 programs

TYPE	Total	Level 0	Level 1
ALL	31	26	5
comments	8	8	0
spacing purposes	8	8	0
textual	0	0	0
assignments	9	9	0
MOVE	8	8	0
ADD	1	1	0
SUBTRACT	0	0	0
COMPUTE	0	0	0
conditionals	2	1	1
IF	0	0	0
ELSE	2	1	1
ON	0	0	0
AT END	0	0	0
branches	7	3	4
CALL	1	1	0
PERFORM	2	1	1
GOTO	4	1	3
NEXT	0	0	0
EXIT	0	0	0
input/output	0	0	0
DELETE	0	0	0
DISPLAY	0	0	0
OPEN	0	0	0
READ	0	0	0
REWRITE	0	0	0
WRITE	0	0	0
labels	4	4	0
other statements	2	2	0
EXAMINE	0	0	0
INSPECT	0	0	0
SEARCH	1	1	0
SET	0	0	0
EXEC CICS	1	1	0
GOBACK	0	0	0

Figure 12. add_sections[TYPE,LEVEL] for revision 6-11 programs

TYPE	all programs	revisions 0-2	revisions 6-11
ALL	0.57	0.60	0.16
comments	0.00	0.00	0.00
spacing purposes	0.00	0.00	0.00
textual	0.00	0.00	0.00
assignments	0.52	0.54	0.00
MOVE	0.52	0.54	0.00
ADD	0.00	0.00	0.00
SUBTRACT	0.00	0.00	0.00
COMPUTE	0.00	0.00	0.00
conditionals	0.62	0.61	0.50
IF	0.00	0.00	0.00
ELSE	1.44	1.48	0.50
ON	0.00	0.00	0.00
AT END	0.00	0.00	0.00
branches	1.06	1.07	0.57
PERFORM	0.81	0.86	0.50
GOTO	0.88	0.84	0.75
NEXT	1.45	1.47	0.00
EXIT	0.00	0.00	0.00
input/output	0.00	0.00	0.00
DELETE	0.00	0.00	0.00
DISPLAY	0.00	0.00	0.00
OPEN	0.00	0.00	0.00
READ	0.00	0.00	0.00
REWRITE	0.00	0.00	0.00
WRITE	0.00	0.00	0.00
labels	0.00	0.00	0.00
other statements	0.00	0.00	0.00
EXAMINE	0.00	0.00	0.00
INSPECT	0.00	0.00	0.00
SEARCH	0.00	0.00	0.00
SET	0.00	0.00	0.00
EXEC CICS	0.00	0.00	0.00
GOBACK	0.00	0.00	0.00

Figure 13. ANL_Add_Sections[TYPE]

TYPE	revisions 0-2	revisions 6-11
ALL	0.263	0.148
comments		
spacing purposes	0.000	0.000
textual	0.000	0.000
assignments		
MOVE	0.179	0.000
ADD	0.000	0.000
SUBTRACT	0.000	0.000
COMPUTE	0.000	0.000
conditionals		
IF	0.000	0.000
ELSE	0.000	0.167
ON	0.000	0.000
AT END	0.000	0.000
branches		
CALL	0.000	0.000
PERFORM	0.000	0.167
GOTO	0.000	0.000
NEXT	0.000	0.000
EXIT	0.000	0.000
input/output		
DELETE	0.000	0.000
DISPLAY	0.000	0.000
OPEN	0.000	0.000
READ	0.000	0.000
REWRITE	0.000	0.000
WRITE	0.000	0.000
labels	0.000	0.000
other statements		
EXAMINE	0.000	0.000
INSPECT	0.000	0.000
SEARCH	0.000	0.000
SET	0.000	0.000
EXEC CICS	0.377	0.000
GOBACK	0.000	0.000

Figure 14. SD_ANL_Add_Sections[TYPE]

TYPE	Adds[TYPE]	%Adds[TYPE]
ALL	4301	
comments	181	4.21
spacing purposes	76	1.77
textual	105	2.44
declarations	212	4.93
sections	0	0.00
declaration	137	3.19
VALUE clauses	71	1.65
redefines	4	0.09
assignments	1188	27.62
MOVE	1182	27.48
ADD	6	0.14
SUBTRACT	0	0.00
COMPUTE	0	0.00
conditionals	1194	27.76
IF	622	14.46
ELSE	546	12.69
ON	2	0.05
AT END	24	0.56
branches	1164	27.06
CALL	4	0.09
PERFORM	198	4.60
GOTO	647	15.04
NEXT	315	7.32
EXIT	0	0.00
input/output	7	0.16
DELETE	1	0.02
DISPLAY	2	0.05
OPEN	0	0.00
READ	1	0.02
REWRITE	0	0.00
WRITE	3	0.07
labels	117	2.72
other statements	238	5.53
EXAMINE	1	0.02
INSPECT	99	2.30
SEARCH	24	0.56
SET	53	1.23
EXEC CICS	57	1.33
GOBACK	4	0.09

Figure 15. Adds[TYPE] and %Adds[TYPE] for all programs

TYPE	Adds[TYPE]	%Adds[TYPE]
ALL	3751	
comments	76	2.03
spacing purposes	28	0.75
textual	48	1.28
declarations	131	3.49
sections	0	0.00
declaration	82	2.19
VALUE clauses	46	1.73
redefines	3	0.08
assignments	1031	27.49
MOVE	1031	27.49
ADD	0	0.00
SUBTRACT	0	0.00
COMPUTE	0	0.00
conditionals	1137	30.31
IF	578	15.41
ELSE	535	14.26
ON	0	0.00
AT END	24	0.64
branches	1094	29.17
CALL	2	0.08
PERFORM	183	4.88
GOTO	603	16.08
NEXT	306	8.16
EXIT	0	0.00
input/output	2	0.08
DELETE	0	0.00
DISPLAY	2	0.08
OPEN	0	0.00
READ	0	0.00
REWRITE	0	0.00
WRITE	0	0.00
labels	84	2.24
other statements	196	5.23
EXAMINE	1	0.03
INSPECT	99	2.64
SEARCH	24	0.64
SET	51	1.36
EXEC CICS	20	0.53
GOBACK	1	0.03

Figure 16. Adds[TYPE] and %Adds[TYPE]
for revision 0-2 programs

TYPE	Adds[TYPE]	%Adds[TYPE]
ALL	260	
comments	28	10.77
spacing purposes	23	8.85
textual	5	1.92
declarations	39	15.00
sections	0	0.00
declaration	29	11.15
VALUE clauses	10	3.85
redefines	0	0.00
assignments	83	31.92
MOVE	81	31.15
ADD	2	0.77
SUBTRACT	0	0.00
COMPUTE	0	0.00
conditionals	30	11.54
IF	24	9.23
ELSE	6	2.31
ON	0	0.00
AT END	0	0.00
branches	33	12.69
CALL	0	0.00
PERFORM	7	2.69
GOTO	20	7.69
NEXT	6	2.31
EXIT	0	0.00
input/output	5	1.92
DELETE	1	0.38
DISPLAY	0	0.00
OPEN	0	0.00
READ	1	0.38
REWRITE	0	0.00
WRITE	3	1.15
labels	21	8.08
other statements	21	8.08
EXAMINE	0	0.00
INSPECT	0	0.00
SEARCH	0	0.00
SET	2	0.77
EXEC CICS	18	6.92
GOBACK	1	0.38

Figure 17. Adds[TYPE] and %Adds[TYPE]
for revision 6-11 programs

TYPE	revisions 0-2	revisions 6-11
comments		
spacing purposes	0.00	0.00
textual	3.78	0.23
declarations		
sections	0.00	0.00
declaration	0.00	0.92
VALUE clauses	0.00	0.00
redefines	0.57	0.00
assignments		
MOVE	0.01	12.97
ADD	0.00	0.03
SUBTRACT	0.00	0.00
COMPUTE	0.00	0.00
conditionals		
IF	0.05	6.93
ELSE	0.12	0.00
ON	0.00	0.00
AT END	0.00	0.10
branches		
CALL	0.03	0.00
PERFORM	0.16	0.00
GOTO	0.00	0.00
NEXT	0.00	1.33
EXIT	0.00	0.00
input/output		
DELETE	0.00	0.00
DISPLAY	0.00	0.42
OPEN	0.00	0.10
READ	0.37	0.00
REWRITE	0.00	0.00
WRITE	0.00	0.00
labels	0.37	1.20
other statements		
EXAMINE	35.60	0.00
INSPECT	0.00	0.00
SEARCH	0.00	0.00
SET	0.00	0.00
EXEC CICS	6.77	5.02
GOBACK	0.00	0.00

Figure 18. SD_#Adds[TYPE]

TYPE	Total	Level 0	Level 1	Level 2
ALL	4022	1327	1542	1153
comments	114	84	30	0
spacing purposes	65	65	0	0
textual	49	19	30	0
assignments	1188	436	710	42
MOVE	1182	432	708	42
ADD	6	4	2	0
SUBTRACT	0	0	0	0
COMPUTE	0	0	0	0
conditionals	1194	424	326	444
IF	622	399	223	0
ELSE	546	0	102	444
ON	2	1	1	0
AT END	24	24	0	0
branches	1164	122	375	667
CALL	4	4	0	0
PERFORM	198	23	175	0
GOTO	647	71	131	445
NEXT	315	24	69	222
EXIT	0	0	0	0
input/output	7	7	0	0
DELETE	1	1	0	0
DISPLAY	2	2	0	0
OPEN	0	0	0	0
READ	1	1	0	0
REWRITE	0	0	0	0
WRITE	3	3	0	0
labels	117	117	0	0
other statements	238	137	101	0
EXAMINE	1	1	0	0
INSPECT	99	0	99	0
SEARCH	24	24	0	0
SET	53	51	2	0
EXEC CICS	57	57	0	0
GOBACK	4	4	0	0

Figure 19. adds[TYPE,LEVEL] for all programs

TYPE	Total	Level 0	Level 1	Level 2
ALL	3574	1037	1385	1152
comments	30	30	0	0
spacing purposes	20	20	0	0
textual	10	10	0	0
assignments	1031	338	651	42
MOVE	1031	338	651	42
ADD	0	0	0	0
SUBTRACT	0	0	0	0
COMPUTE	0	0	0	0
conditionals	1137	380	313	444
IF	578	356	222	0
ELSE	535	0	91	444
ON	0	0	0	0
AT END	24	24	0	0
branches	1094	107	321	666
CALL	2	2	0	0
PERFORM	183	15	168	0
GOTO	603	66	93	444
NEXT	306	24	60	222
EXIT	0	0	0	0
input/output	2	2	0	0
DELETE	0	0	0	0
DISPLAY	2	2	0	0
OPEN	0	0	0	0
READ	0	0	0	0
REWRITE	0	0	0	0
WRITE	0	0	0	0
labels	84	84	0	0
other statements	196	96	100	0
EXAMINE	1	1	0	0
INSPECT	99	0	99	0
SEARCH	24	24	0	0
SET	51	50	1	0
EXEC CICS	20	20	0	0
GOBACK	1	1	0	0

Figure 20. adds[TYPE,LEVEL] for revision 0-2 programs

TYPE	Total	Level 0	Level 1	Level 2
ALL	221	165	55	1
comments	28	27	1	0
spacing purposes	23	23	0	0
textual	5	4	1	0
assignments	83	63	20	0
MOVE	81	61	20	0
ADD	2	2	0	0
SUBTRACT	0	0	0	0
COMPUTE	0	0	0	0
conditionals	30	23	7	0
IF	24	23	1	0
ELSE	6	0	6	0
ON	0	0	0	0
AT END	0	0	0	0
branches	33	6	26	1
CALL	0	0	0	0
PERFORM	7	4	3	0
GOTO	20	2	17	1
NEXT	6	0	6	0
EXIT	0	0	0	0
input/output	5	5	0	0
DELETE	1	1	0	0
DISPLAY	0	0	0	0
OPEN	0	0	0	0
READ	1	1	0	0
REWRITE	0	0	0	0
WRITE	3	3	0	0
labels	21	21	0	0
other statements	21	20	1	0
EXAMINE	0	0	0	0
INSPECT	0	0	0	0
SEARCH	0	0	0	0
SET	2	1	1	0
EXEC CICS	18	18	0	0
GOBACK	1	1	0	0

Figure 21. adds[TYPE,LEVEL] for revision 6-11 programs

TYPE	all programs	revisions 0-2	revisions 6-11
ALL	0.96	1.03	0.51
comments	0.26	0.00	0.04
spacing purposes	0.00	0.00	0.00
textual	0.61	0.00	0.20
assignments	0.67	0.71	0.24
MOVE	0.67	0.71	0.25
ADD	0.33	0.00	0.00
SUBTRACT	0.00	0.00	0.00
COMPUTE	0.00	0.00	0.00
conditionals	1.02	1.06	0.23
IF	0.36	0.38	0.04
ELSE	1.81	1.83	1.00
ON	0.50	0.00	0.00
AT END	0.00	0.00	0.00
branches	1.47	1.51	0.85
PERFORM	0.88	0.92	0.43
GOTO	1.58	1.63	0.95
NEXT	1.63	1.65	0.00
EXIT	0.00	0.00	0.00
input/output	0.00	0.00	0.00
DELETE	0.00	0.00	0.00
DISPLAY	0.00	0.00	0.00
OPEN	0.00	0.00	0.00
READ	0.00	0.00	0.00
REWRITE	0.00	0.00	0.00
WRITE	0.00	0.00	0.00
labels	0.00	0.00	0.00
other statements	0.00	0.00	0.00
EXAMINE	0.00	0.00	0.00
INSPECT	0.00	0.00	0.00
SEARCH	0.00	0.00	0.00
SET	0.00	0.00	0.00
EXEC CICS	0.00	0.00	0.00
GOBACK	0.00	0.00	0.00

Figure 22. ANL_Adds[TYPE]

TYPE	revisions 0-2	revisions 6-11
ALL	0.512	0.000
comments		
spacing purposes	0.000	0.000
textual	0.000	0.000
assignments		
MOVE	0.630	0.230
ADD	0.000	0.000
SUBTRACT	0.000	0.000
COMPUTE	0.000	0.000
conditionals		
IF	0.000	0.089
ELSE	0.337	0.000
ON	0.000	0.084
AT END	0.000	0.000
branches		
CALL	0.000	0.000
PERFORM	0.346	0.000
GOTO	0.000	0.000
NEXT	0.350	0.000
EXIT	0.000	0.000
input/output		
DELETE	0.000	0.000
DISPLAY	0.000	0.000
OPEN	0.000	0.000
READ	0.000	0.000
REWRITE	0.000	0.000
WRITE	0.000	0.000
labels	0.000	0.000
other statements		
EXAMINE	0.000	0.000
INSPECT	0.000	0.000
SEARCH	0.000	0.000
SET	0.000	0.167
EXEC CICS	0.000	0.000
GOBACK	0.000	0.000

Figure 23. SD_ANL_Adds[TYPE]

TYPE	Changes[TYPE]	%Changes[TYPE]	Changes%[TYPE]
ALL	3497		2.22
comments	73	2.09	0.39
spacing purposes	20	0.57	0.16
textual	53	1.52	0.93
declarations	142	4.06	0.39
sections	0	0.00	0.00
declaration	116	3.32	0.38
VALUE clauses	26	0.74	0.45
redefines	0	0.00	0.00
assignments	2219	63.45	4.19
MOVE	2209	63.17	4.29
ADD	10	0.29	0.71
SUBTRACT	0	0.00	0.00
COMPUTE	0	0.00	0.00
conditionals	437	12.50	2.01
IF	266	7.61	1.71
ELSE	167	4.78	2.79
ON	4	0.11	5.13
AT END	0	0.00	0.00
branches	376	10.75	2.18
CALL	9	0.26	2.10
PERFORM	180	5.15	3.55
GOTO	146	4.18	1.49
NEXT	41	1.17	1.71
EXIT	0	0.00	0.00
input/output	4	0.11	0.84
DELETE	1	0.03	1.15
DISPLAY	0	0.00	0.00
OPEN	1	0.03	5.88
READ	0	0.00	0.00
REWRITE	2	0.06	33.33
WRITE	0	0.00	0.00
labels	16	0.46	0.26
other statements	230	6.58	6.02
EXAMINE	135	3.86	22.88
INSPECT	0	0.00	0.00
SEARCH	1	0.03	0.44
SET	11	0.31	4.58
EXEC CICS	83	2.37	3.36
GOBACK	0	0.00	0.00

Figure 24. Changes[TYPE], %Changes[TYPE], and Changes%[TYPE]
for all programs

TYPE	Changes[TYPE]	%Changes[TYPE]	Changes%[TYPE]
ALL	2465		2.45
comments	21	0.85	2.42
spacing purposes	11	0.45	0.19
textual	10	0.41	0.34
declarations	90	3.65	0.44
sections	0	0.00	0.00
declaration	77	3.12	0.47
VALUE clauses	13	0.53	0.35
redefines	0	0.00	0.00
assignments	1635	66.33	4.47
MOVE	1633	66.25	4.60
ADD	2	0.08	0.19
SUBTRACT	0	0.00	0.00
COMPUTE	0	0.00	0.00
conditionals	258	10.47	1.70
IF	138	5.60	1.31
ELSE	118	4.79	2.65
ON	2	0.08	4.76
AT END	0	0.00	0.00
branches	262	10.63	2.13
CALL	5	0.20	2.66
PERFORM	144	5.84	4.64
GOTO	86	3.49	1.26
NEXT	27	1.10	1.24
EXIT	0	0.00	0.00
input/output	2	0.08	0.54
DELETE	0	0.00	0.00
DISPLAY	0	0.00	0.00
OPEN	1	0.04	10.00
READ	0	0.00	0.00
REWRITE	1	0.04	33.33
WRITE	0	0.00	0.00
labels	2	0.08	0.05
other statements	195	7.91	6.39
EXAMINE	134	5.44	23.34
INSPECT	0	0.00	0.00
SEARCH	0	0.00	0.00
SET	8	0.32	3.88
EXEC CICS	53	2.15	2.93
GOBACK	0	0.00	0.00

Figure 25. Changes[TYPE], %Changes[TYPE], and Changes%[TYPE]
for revisions 0-2 programs

TYPE	Changes[TYPE]	%Changes[TYPE]	Changes%[TYPE]
ALL	288		0.83
comments	2	0.69	0.03
spacing purposes	2	0.69	0.04
textual	0	0.00	0.00
declarations	11	3.82	0.13
sections	0	0.00	0.00
declaration	10	3.47	0.13
VALUE clauses	2	0.69	0.21
redefines	0	0.00	0.00
assignments	199	69.10	1.89
MOVE	198	68.75	1.91
ADD	1	0.35	0.57
SUBTRACT	0	0.00	0.00
COMPUTE	0	0.00	0.00
conditionals	29	10.07	0.61
IF	19	6.60	0.53
ELSE	10	3.47	0.88
ON	0	0.00	0.00
AT END	0	0.00	0.00
branches	32	11.11	1.03
CALL	2	0.69	1.07
PERFORM	10	3.47	0.73
GOTO	16	5.56	1.07
NEXT	4	1.39	11.11
EXIT	0	0.00	0.00
input/output	2	0.69	5.56
DELETE	1	0.35	25.00
DISPLAY	0	0.00	0.00
OPEN	0	0.00	0.00
READ	0	0.00	0.00
REWRITE	1	0.35	100.00
WRITE	0	0.00	0.00
labels	3	1.04	0.24
other statements	10	3.47	3.42
EXAMINE	0	0.00	0.00
INSPECT	0	0.00	0.00
SEARCH	0	0.00	0.00
SET	1	0.35	20.00
EXEC CICS	9	3.13	6.87
GOBACK	0	0.00	0.00

Figure 26. Changes[TYPE], %Changes[TYPE], and Changes%[TYPE]
for revision 6-11 programs

TYPE	revisions 0-2	revisions 6-11
comments		
spacing purposes	4.53	0.71
textual	0.00	0.00
declarations		
sections	0.00	0.00
declaration	1.64	0.65
VALUE clauses	0.00	0.65
redefines	0.00	0.00
assignments		
MOVE	0.00	0.00
ADD	0.00	0.00
SUBTRACT	0.00	0.00
COMPUTE	0.00	0.00
conditionals		
IF	7.25	0.00
ELSE	15.40	0.00
ON	0.00	0.00
AT END	0.00	0.00
branches		
CALL	9.93	0.00
PERFORM	0.00	5.20
GOTO	0.00	5.10
NEXT	7.95	0.00
EXIT	0.00	0.00
input/output		
DELETE	0.00	0.00
DISPLAY	0.00	0.00
OPEN	0.00	0.00
READ	0.00	0.00
REWRITE	0.00	0.00
WRITE	0.00	0.00
labels	0.37	1.20
other statements		
EXAMINE	35.60	0.00
INSPECT	0.00	0.00
SEARCH	0.00	0.00
SET	0.00	0.00
EXEC CICS	6.77	5.02
GOBACK	0.00	0.00

Figure 27. SD_3Changes[TYPE]

TYPE	Total	Level 0	Level 1	Level 2
ALL	3344	715	2127	432
comments	62	24	38	0
spacing purposes	11	11	0	0
textual	51	13	38	0
assignments	2219	379	1592	198
MOVE	2209	377	1585	197
ADD	10	2	7	1
SUBTRACT	0	0	0	0
COMPUTE	0	0	0	0
conditionals	437	145	135	150
IF	266	144	95	25
ELSE	167	1	36	125
ON	4	0	4	0
AT END	0	0	0	0
branches	376	57	222	84
CALL	9	9	0	0
PERFORM	180	12	146	17
GOTO	146	36	54	51
NEXT	41	0	22	16
EXIT	0	0	0	0
input/output	4	3	1	0
DELETE	1	1	0	0
DISPLAY	0	0	0	0
OPEN	1	0	1	0
READ	0	0	0	0
REWRITE	2	2	0	0
WRITE	0	0	0	0
labels	16	16	0	0
other statements	230	91	139	0
EXAMINE	135	7	128	0
INSPECT	0	0	0	0
SEARCH	1	1	0	0
SET	11	11	0	0
EXEC CICS	83	72	11	0
GOBACK	0	0	0	0

Figure 28a. changes[TYPE,LEVEL] for all programs (Levels 0-2)

TYPE	Level 3	Level 4	Level 5
ALL	52	3	15
comments	0	0	0
spacing purposes	0	0	0
textual	0	0	0
assignments	48	0	2
MOVE	48	0	2
ADD	0	0	0
SUBTRACT	0	0	0
COMPUTE	0	0	0
conditionals	1	1	5
IF	1	1	0
ELSE	0	0	5
ON	0	0	0
AT END	0	0	0
branches	3	2	8
CALL	0	0	0
PERFORM	2	1	2
GOTO	1	1	3
NEXT	0	0	3
EXIT	0	0	0
input/output	0	0	0
DELETE	0	0	0
DISPLAY	0	0	0
OPEN	0	0	0
READ	0	0	0
REWRITE	0	0	0
WRITE	0	0	0
labels	0	0	0
other statements	0	0	0
EXAMINE	0	0	0
INSPECT	0	0	0
SEARCH	0	0	0
SET	0	0	0
EXEC CICS	0	0	0
GOBACK	0	0	0

Figure 28b. changes[TYPE,LEVEL] for all programs (Levels 3-5)

TYPE	Total	Level 0	Level 1	Level 2
ALL	2368	401	1635	299
comments	14	14	0	0
spacing purposes	5	5	0	0
textual	9	9	0	0
assignments	1635	239	1233	131
MOVE	1633	239	1232	130
ADD	2	0	1	1
SUBTRACT	0	0	0	0
COMPUTE	0	0	0	0
conditionals	258	48	102	108
IF	138	47	75	16
ELSE	118	1	25	92
ON	2	0	2	0
AT END	0	0	0	0
branches	262	40	161	60
CALL	5	5	0	0
PERFORM	144	6	126	11
GOTO	86	29	18	39
NEXT	27	0	17	10
EXIT	0	0	0	0
input/output	2	1	1	0
DELETE	0	0	0	0
DISLAY	0	0	0	0
OPEN	1	0	1	0
READ	0	0	0	0
REWRITE	1	1	0	0
WRITE	0	0	0	0
labels	2	2	0	0
other statements	195	57	138	0
EXAMINE	134	6	128	0
INSPECT	0	0	0	0
SEARCH	0	0	0	0
SET	8	8	0	0
EXEC CICS	53	43	10	0
GOBACK	0	0	0	0

Figure 29. changes[TYPE,LEVEL] for revision 0-2 programs

TYPE	Total	Level 0	Level 1	Level 2
ALL	275	77	164	13
comments	0	0	0	0
spacing purposes	0	0	0	0
textual	0	0	0	0
assignments	199	42	149	6
MOVE	198	41	149	6
ADD	1	1	0	0
SUBTRACT	0	0	0	0
COMPUTE	0	0	0	0
conditionals	29	14	5	3
IF	19	14	2	1
ELSE	10	0	3	2
ON	0	0	0	0
AT END	0	0	0	0
branches	32	7	9	4
CALL	2	2	0	0
PERFORM	10	3	3	0
GOTO	16	2	5	4
NEXT	4	0	1	0
EXIT	0	0	0	0
input/output	2	2	0	0
DELETE	1	1	0	0
DISPLAY	0	0	0	0
OPEN	0	0	0	0
READ	0	0	0	0
REWRITE	1	1	0	0
WRITE	0	0	0	0
labels	3	3	0	0
other statements	10	9	1	0
EXAMINE	0	0	0	0
INSPECT	0	0	0	0
SEARCH	0	0	0	0
SET	1	1	0	0
EXEC CICS	9	8	1	0
GOBACK	0	0	0	0

Figure 30a. changes[TYPE,LEVEL] for revision 6-11 programs (Levels 0-2)

TYPE	Level 3	Level 4	Level 5
ALL	3	3	15
comments	0	0	0
spacing purposes	0	0	0
textual	0	0	0
assignments	0	0	2
MOVE	0	0	2
ADD	0	0	0
SUBTRACT	0	0	0
COMPUTE	0	0	0
conditionals	1	1	5
IF	1	1	0
ELSE	0	0	5
ON	0	0	0
AT END	0	0	0
branches	2	2	8
CALL	0	0	0
PERFORM	1	1	2
GOTO	1	1	3
NEXT	0	0	3
EXIT	0	0	0
input/output	0	0	0
DELETE	0	0	0
DISPLAY	0	0	0
OPEN	0	0	0
READ	0	0	0
REWRITE	0	0	0
WRITE	0	0	0
labels	0	0	0
other statements	0	0	0
EXAMINE	0	0	0
INSPECT	0	0	0
SEARCH	0	0	0
SET	0	0	0
EXEC CICS	0	0	0
GOBACK	0	0	0

Figure 30b. changes[TYPE,LEVEL] for revision 6-11 programs (Levels 3-5)

TYPE	all programs	revisions 0-2	revisions 6-11
ALL	0.97	0.98	1.04
comments	0.61	0.00	0.00
spacing purposes	0.00	0.00	0.00
textual	0.75	0.00	0.00
assignments	0.97	0.95	0.86
MOVE	0.97	0.95	0.86
ADD	0.90	1.50	0.00
SUBTRACT	0.00	0.00	0.00
COMPUTE	0.00	0.00	0.00
conditionals	1.07	1.23	1.48
IF	0.57	0.78	0.58
ELSE	1.86	1.77	3.20
ON	1.00	1.00	0.00
AT END	0.00	0.00	0.00
branches	1.19	1.08	2.22
PERFORM	1.11	1.05	2.00
GOTO	1.22	1.12	2.19
NEXT	1.68	1.37	4.00
EXIT	0.00	0.00	0.00
input/output	0.25	0.50	0.00
DELETE	0.00	0.00	0.00
DISPLAY	0.00	0.00	0.00
OPEN	1.00	1.00	0.00
READ	0.00	0.00	0.00
REWRITE	0.00	0.00	0.00
WRITE	0.00	0.00	0.00
labels	0.00	0.00	0.00
other statements	0.60	0.71	0.10
EXAMINE	0.95	0.96	0.00
INSPECT	0.00	0.00	0.00
SEARCH	0.00	0.00	0.00
SET	0.00	0.00	0.00
EXEC CICS	0.13	0.19	0.11
GOBACK	0.00	0.00	0.00

Figure 31. ANL_Changes[TYPE]

TYPE	revisions 0-2	revisions 6-11
ALL	0.000	0.424
comments		
spacing purposes	0.000	0.000
textual	0.000	0.000
assignments		
MOVE	0.468	0.338
ADD	0.000	0.000
SUBTRACT	0.000	0.000
COMPUTE	0.000	0.000
conditionals		
IF	0.468	0.134
ELSE	0.000	0.000
ON	0.000	0.000
AT END	0.000	0.000
branches		
CALL	0.000	0.000
PERFORM	0.000	0.425
GOTO	0.000	0.000
NEXT	0.000	0.000
EXIT	0.000	0.000
input/output		
DELETE	0.000	0.000
DISPLAY	0.000	0.000
OPEN	0.232	0.000
READ	0.000	0.000
REWRITE	0.000	0.000
WRITE	0.000	0.000
labels	0.000	0.000
other statements		
EXAMINE	0.000	0.000
INSPECT	0.000	0.000
SEARCH	0.000	0.000
SET	0.000	0.000
EXEC CICS	0.000	0.000
GOBACK	0.000	0.000

Figure 32. SD_ANL_Changes[TYPE] for all programs

TYPE1	TYPE2	CHANGES[TYPE1,TYPE2]	CHANGES%[TYPE1,TYPE2]
Comment	Comment	25	34.25
	Procedure Division Statement	48	65.75
Declaration	Declaration	98	84.98
	Comment	18	15.52
VALUE Clause	VALUE Clause	23	88.46
	Comment	3	11.54
MOVE	MOVE	1519	68.76
	IF MOVE ELSE MOVE	232	10.50
	IF MOVE	36	1.63
	ELSE MOVE	20	0.91
	Comment	402	18.20
ADD	ADD	2	20.00
	Comment	8	80.00
IF	IF	185	69.55
	MOVE	3	1.13
	Comment	78	29.32
ELSE	ELSE	86	51.50
	Comment	81	48.50
ON	Comment	4	100.00
CALL	CALL	3	33.33
	Comment	6	66.66
PERFORM	PERFORM	87	48.33
	IF PERFORM	6	3.33
	GO TO	19	10.56
	Comment	68	37.78
GO TO	GO TO	85	58.22
	IF GO TO	7	4.79
	PERFORM	3	2.06
	Comment	51	34.93
NEXT	NEXT	7	17.07
	PERFORM	5	12.20
	GO TO	17	41.46
	Comment	12	29.27
DELETE	DELETE	1	100.00
OPEN	OPEN	1	100.00

REWRITE	REWRITE	2	100.00
LABEL	LABEL	5	31.25
	Comment	11	68.75
EXAMINE	EXAMINE	39	28.89
	IF EXAMINE	6	4.45
	MOVE	5	3.70
	Comment	85	62.96
SEARCH	Comment	1	100.00
SET	SET	2	18.18
	Comment	9	81.82
EXEC CICS	EXEC CICS	59	71.08
	IF EXEC CICS	3	3.62
	Comment	21	25.30

Figure 33. CHANGES[TYPE1,TYPE2] and CHANGES%(TYPE1,TYPE2] for all programs

TYPE	Deletes[TYPE]	%Deletes[TYPE]	Deletes%[TYPE]
ALL	2022		1.28
comments	330	16.32	1.78
spacing purposes	103	5.09	0.80
textual	227	11.23	3.97
declarations	661	32.69	1.80
sections	0	0.00	0.00
declaration	545	26.95	1.80
VALUE clauses	93	4.60	1.63
redefines	23	1.14	3.81
assignments	565	27.94	1.07
MOVE	565	27.94	1.10
ADD	0	0.00	0.00
SUBTRACT	0	0.00	0.00
COMPUTE	0	0.00	0.00
conditionals	262	12.96	1.20
IF	210	10.39	1.35
ELSE	50	2.47	0.84
ON	0	0.00	0.00
AT END	2	0.10	1.20
branches	149	7.37	0.87
CALL	2	0.10	0.47
PERFORM	36	1.78	0.71
GOTO	80	3.96	0.82
NEXT	31	1.53	1.29
EXIT	0	0.00	0.00
input/output	2	0.10	0.42
DELETE	0	0.00	0.00
DISPLAY	1	0.05	14.29
OPEN	1	0.05	5.88
READ	0	0.00	0.00
REWRITE	0	0.00	0.00
WRITE	0	0.00	0.00
labels	34	1.68	0.56
other statements	19	0.94	0.50
EXAMINE	0	0.00	0.00
INSPECT	0	0.00	0.00
SEARCH	2	0.10	0.88
SET	2	0.10	0.83
EXEC CICS	15	0.74	0.61
GOBACK	0	0.00	0.00

Figure 34. Deletes[TYPE], %Deletes[TYPE], and Deletes%[TYPE]
for all programs

TYPE	Deletes[TYPE]	%Deletes[TYPE]	Deletes%[TYPE]
ALL	1047		1.04
comments	295	28.18	3.04
spacing purposes	87	8.31	1.53
textual	208	19.87	6.97
declarations	298	28.46	1.45
sections	0	0.00	0.00
declaration	235	22.45	1.43
VALUE clauses	60	5.73	1.62
redefines	3	0.29	0.92
assignments	267	25.50	0.73
MOVE	267	25.50	0.75
ADD	0	0.00	0.00
SUBTRACT	0	0.00	0.00
COMPUTE	0	0.00	0.00
conditionals	118	11.27	0.78
IF	95	9.07	0.90
ELSE	23	2.20	0.52
ON	0	0.00	0.00
AT END	0	0.00	0.00
branches	44	4.20	0.36
CALL	1	0.10	0.53
PERFORM	3	0.29	0.10
GOTO	24	2.29	0.35
NEXT	16	1.53	0.73
EXIT	0	0.00	0.00
input/output	1	0.10	0.27
DELETE	0	0.00	0.00
DISPLAY	1	0.10	20.00
OPEN	0	0.00	0.00
READ	0	0.00	0.00
REWRITE	0	0.00	0.00
WRITE	0	0.00	0.00
labels	12	1.15	0.30
other statements	12	1.15	0.39
EXAMINE	0	0.00	0.00
INSPECT	0	0.00	0.00
SEARCH	0	0.00	0.00
SET	0	0.00	0.00
EXEC CICS	12	1.15	0.66
GOBACK	0	0.00	0.00

Figure 35. Deletes[TYPE], %Deletes[TYPE], and Deletes%[TYPE]
for revision 0-2 programs

TYPE	Deletes[TYPE]	%Deletes[TYPE]	Deletes%[TYPE]
ALL	170		0.49
comments	8	4.71	0.13
spacing purposes	8	4.71	0.18
textual	0	0.00	0.00
declarations	16	9.41	0.18
sections	5	2.94	45.45
declaration	7	4.12	0.09
VALUE clauses	4	2.35	0.42
redefines	0	0.00	0.00
assignments	115	67.65	1.07
MOVE	115	67.65	1.09
ADD	0	0.00	0.00
SUBTRACT	0	0.00	0.00
COMPUTE	0	0.00	0.00
conditionals	22	12.94	0.46
IF	21	12.35	0.59
ELSE	1	0.59	0.09
ON	0	0.00	0.00
AT END	0	0.00	0.00
branches	4	2.35	0.13
CALL	0	0.00	0.00
PERFORM	1	0.59	0.70
GOTO	3	1.76	0.74
NEXT	0	0.00	0.00
EXIT	0	0.00	0.00
input/output	1	0.59	2.78
DELETE	0	0.00	0.00
DISPLAY	0	0.00	0.00
OPEN	1	0.59	100.00
READ	0	0.00	0.00
REWRITE	0	0.00	0.00
WRITE	0	0.00	0.00
labels	1	0.59	0.08
other statements	3	1.76	1.76
EXAMINE	0	0.00	0.00
INSPECT	0	0.00	0.00
SEARCH	0	0.00	0.00
SET	0	0.00	0.00
EXEC CICS	3	1.76	1.76
GOBACK	0	0.00	0.00

Figure 36. Deletes[TYPE], %Deletes[TYPE], and Deletes%[TYPE]
for revision 6-11 programs

TYPE	revisions 0-2	revisions 6-11
comments		
spacing purposes	0.00	1.42
textual	0.00	0.00
declarations		
sections	0.00	0.00
declaration	0.00	0.57
VALUE clauses	0.00	1.30
redefines	11.71	0.00
assignments		
MOVE	8.69	0.00
ADD	0.00	0.00
SUBTRACT	0.00	0.00
COMPUTE	0.00	0.00
conditionals		
IF	0.00	0.00
ELSE	0.00	0.00
ON	0.00	0.00
AT END	0.00	0.00
branches		
CALL	0.00	0.00
PERFORM	0.00	0.65
GOTO	1.32	0.00
NEXT	0.00	0.00
EXIT	0.00	0.00
input/output		
DELETE	0.00	0.00
DISPLAY	0.00	0.00
OPEN	0.00	0.00
READ	0.00	0.00
REWRITE	0.00	0.00
WRITE	0.00	0.00
labels	0.00	0.00
other statements		
EXAMINE	0.00	0.00
INSPECT	0.00	0.00
SEARCH	0.00	0.00
SET	0.00	0.00
EXEC CICS	6.99	3.06
GOBACK	0.00	0.00

Figure 37. SD_Deletes[TYPE]

TYPE	Total	Level 0	Level 1	Level 2
ALL	1304	610	579	97
comments	273	273	0	0
spacing purposes	66	66	0	0
textual	207	207	0	0
assignments	565	80	464	21
MOVE	565	80	464	21
ADD	0	0	0	0
SUBTRACT	0	0	0	0
COMPUTE	0	0	0	0
conditionals	262	192	30	33
IF	210	190	17	1
ELSE	50	0	13	32
ON	0	0	0	0
AT END	2	2	0	0
branches	149	14	81	43
CALL	2	2	0	0
PERFORM	36	2	22	9
GOTO	80	10	47	18
NEXT	31	0	12	16
EXIT	0	0	0	0
input/output	2	0	2	0
DELETE	0	0	0	0
DISPLAY	1	0	1	0
OPEN	1	0	1	0
READ	0	0	0	0
REWRITE	0	0	0	0
WRITE	0	0	0	0
labels	34	34	0	0
other statements	19	17	2	0
EXAMINE	0	0	0	0
INSPECT	0	0	0	0
SEARCH	2	2	0	0
SET	2	2	0	0
EXEC CICS	15	13	2	0
GOBACK	0	0	0	0

Figure 38a. deletes[TYPE,LEVEL] for all programs (Levels 0-2)

TYPE	Level 3	Level 4	Level 5
ALL	3	3	12
comments	0	0	0
spacing purposes	0	0	0
textual	0	0	0
assignments	0	0	0
MOVE	0	0	0
ADD	0	0	0
SUBTRACT	0	0	0
COMPUTE	0	0	0
conditionals	1	1	5
IF	1	1	0
ELSE	0	0	5
ON	0	0	0
AT END	0	0	0
branches	2	2	7
CALL	0	0	0
PERFORM	1	1	1
GOTO	1	1	3
NEXT	0	0	3
EXIT	0	0	0
input/output	0	0	0
DELETE	0	0	0
DISPLAY	0	0	0
OPEN	0	0	0
READ	0	0	0
REWRITE	0	0	0
WRITE	0	0	0
labels	0	0	0
other statements	0	0	0
EXAMINE	0	0	0
INSPECT	0	0	0
SEARCH	0	0	0
SET	0	0	0
EXEC CICS	0	0	0
GOBACK	0	0	0

Figure 38b. deletes[TYPE,LEVEL] for all programs (Levels 3 - 5)

TYPE	Total	Level 0	Level 1	Level 2
ALL	715	424	249	42
comments	261	261	0	0
spacing purposes	56	56	0	0
textual	205	205	0	0
assignments	267	48	198	21
MOVE	267	48	198	21
ADD	0	0	0	0
SUBTRACT	0	0	0	0
COMPUTE	0	0	0	0
conditionals	118	88	16	14
IF	95	88	7	0
ELSE	23	0	9	14
ON	0	0	0	0
AT END	0	0	0	0
branches	44	3	34	7
CALL	1	1	0	0
PERFORM	3	1	2	0
GOTO	24	1	23	0
NEXT	16	0	9	7
EXIT	0	0	0	0
input/output	1	0	1	0
DELETE	0	0	0	0
DISPLAY	1	0	1	0
OPEN	0	0	0	0
READ	0	0	0	0
REWRITE	0	0	0	0
WRITE	0	0	0	0
labels	12	12	0	0
other statements	12	12	0	0
EXAMINE	0	0	0	0
INSPECT	0	0	0	0
SEARCH	0	0	0	0
SET	0	0	0	0
EXEC CICS	12	12	0	0
GOBACK	0	0	0	0

Figure 39. deletes[TYPE,LEVEL] for revision 0-2 programs

TYPE	Total	Level 0	Level 1
ALL	154	44	110
comments	8	8	0
spacing purposes	8	8	0
textual	0	0	0
assignments	115	11	104
MOVE	115	11	104
ADD	0	0	0
SUBTRACT	0	0	0
COMPUTE	0	0	0
conditionals	22	21	1
IF	21	21	0
ELSE	1	0	1
ON	0	0	0
AT END	0	0	0
branches	4	2	2
CALL	0	0	0
PERFORM	1	1	0
GOTO	3	1	2
NEXT	0	0	0
EXIT	0	0	0
input/output	1	0	1
DELETE	0	0	0
DISPLAY	0	0	0
OPEN	1	0	1
READ	0	0	0
REWRITE	0	0	0
WRITE	0	0	0
labels	1	1	0
other statements	3	1	2
EXAMINE	0	0	0
INSPECT	0	0	0
SEARCH	0	0	0
SET	0	0	0
EXEC CICS	3	1	2
GOBACK	0	0	0

Figure 40. deletes[TYPE,LEVEL] for revision 6-11 programs

TYPE	all programs	revisions 0-2	revisions 6-11
ALL	0.65	0.47	0.71
comments	0.00	0.00	0.00
spacing purposes	0.00	0.00	0.00
textual	0.00	0.00	0.00
assignments	0.90	0.90	0.90
MOVE	0.90	0.90	0.90
ADD	0.00	0.00	0.00
SUBTRACT	0.00	0.00	0.00
COMPUTE	0.00	0.00	0.00
conditionals	0.49	0.37	0.05
IF	0.12	0.07	0.00
ELSE	2.04	1.61	1.00
ON	0.00	0.00	0.00
AT END	0.00	0.00	0.00
branches	1.45	1.09	0.50
PERFORM	1.44	0.66	0.00
GOTO	1.31	0.96	0.66
NEXT	1.90	1.44	0.00
EXIT	0.00	0.00	0.00
input/output	1.00	1.00	1.00
DELETE	0.00	0.00	0.00
DISPLAY	1.00	1.00	0.00
OPEN	1.00	0.00	1.00
READ	0.00	0.00	0.00
REWRITE	0.00	0.00	0.00
WRITE	0.00	0.00	0.00
labels	0.00	0.00	0.00
other statements	0.11	0.00	0.66
EXAMINE	0.00	0.00	0.00
INSPECT	0.00	0.00	0.00
SEARCH	0.00	0.00	0.00
SET	0.00	0.00	0.00
EXEC CICS	0.13	0.00	0.66
GOBACK	0.00	0.00	0.00

Figure 41. ANL_Deletes[TYPE]

TYPE	revisions 0-2	revisions 6-11
ALL	0.305	0.000
comments		
spacing purposes	0.000	0.000
textual	0.000	0.000
assignments		
MOVE	0.543	0.290
ADD	0.000	0.000
SUBTRACT	0.000	0.000
COMPUTE	0.000	0.000
conditionals		
IF	0.000	0.000
ELSE	0.648	0.167
ON	0.000	0.000
AT END	0.000	0.000
branches		
CALL	0.000	0.000
PERFORM	0.000	0.000
GOTO	0.416	0.232
NEXT	0.648	0.000
EXIT	0.000	0.000
input/output		
DELETE	0.000	0.000
DISPLAY	0.167	0.000
OPEN	0.000	0.167
READ	0.000	0.000
REWRITE	0.000	0.000
WRITE	0.000	0.000
labels	0.000	0.000
other statements		
EXAMINE	0.000	0.000
INSPECT	0.000	0.000
SEARCH	0.000	0.000
SET	0.000	0.000
EXEC CICS	0.377	0.000
GOBACK	0.000	0.000

Figure 42. SD_ANL_Deletes[TYPE]

Number of IF statements nested to a depth of	Level 1	:	8764
	Level 2	:	2412
	Level 3	:	393
	Level 4	:	102
	Level 5	:	27
	Level 6	:	10
	Level 7	:	25

Figure 43. Depth of nesting of IF statements

		CHANGES to IF statements							
Deepest level of changed IF statement	7	0	0	0	0	0	0	0	0
	6	0	0	0	0	0	0	0	-
	5	6	1	1	1	1	0	-	-
	4	6	0	1	0	0	-	-	-
	3	13	39	23	0	-	-	-	-
	2	17	55	0	-	-	-	-	-
	1	102	0	-	-	-	-	-	-
	0	0	-	-	-	-	-	-	-
		0	1	2	3	4	5	6	7
		Nesting level of changed IF statement							

Figure 44. Environment of changed IF statements

		CHANGES to ELSE statements							
Deepest level of changed ELSE statement	7	0	0	0	0	0	0	0	0
	6	0	0	0	0	0	0	0	-
	5	0	1	15	0	0	5	-	-
	4	0	0	37	0	0	-	-	-
	3	0	0	59	0	-	-	-	-
	2	0	3	14	-	-	-	-	-
	1	0	32	-	-	-	-	-	-
	0	1	-	-	-	-	-	-	-
		0	1	2	3	4	5	6	7
		Nesting level of changed ELSE statement							

Figure 45. Environment of changed ELSE statements

		DELETIONS of IF statements							
Deepest level of deleted IF statement	7	0	0	0	0	0	0	0	0
	6	0	0	0	0	0	0	0	-
	5	10	1	1	1	1	0	-	-
	4	16	0	0	0	0	-	-	-
	3	17	3	0	0	-	-	-	-
	2	19	13	0	-	-	-	-	-
	1	128	0	-	-	-	-	-	-
	0	0	-	-	-	-	-	-	-
		0	1	2	3	4	5	6	7
		Nesting level of deleted IF statement							

Figure 46. Environment of deleted IF statements

		DELETIONS of ELSE statements							
Deepest level of deleted ELSE statement	7	0	0	0	0	0	0	0	0
	6	0	0	0	0	0	0	0	-
	5	0	0	2	0	0	5	-	-
	4	0	0	0	0	0	-	-	-
	3	0	0	13	0	-	-	-	-
	2	0	13	17	-	-	-	-	-
	1	0	0	-	-	-	-	-	-
	0	0	-	-	-	-	-	-	-
		0	1	2	3	4	5	6	7
		Nesting level of deleted ELSE statement							

Figure 47. Environment of deleted ELSE statements

		Percent of CHANGES to IF statements					
Deepest level of changed IF statement	7	0.00	0.00	0.00	0.00	0.00	0.00
	6	0.00	0.00	0.00	0.00	0.00	0.00
	5	22.22	3.70	3.70	3.70	3.70	0.00
	4	5.88	0.00	0.98	0.00	0.00	-
	3	3.31	9.92	5.85	0.00	-	-
	2	0.70	2.28	0.00	-	-	-
	1	1.16	0.00	-	-	-	-
	0	0	-	-	-	-	-
		0	1	2	3	4	5
		Nesting level of changed IF statement					

Figure 48. Enviroment of changed IF statements (percentages)

		Percent of CHANGES to ELSE statements					
Deepest level of changed ELSE statement	7	0.00	0.00	0.00	0.00	0.00	0.00
	6	0.00	0.00	0.00	0.00	0.00	0.00
	5	0.00	5.26	7.89	0.00	0.00	26.32
	4	0.00	0.00	66.07	0.00	0.00	-
	3	0.00	0.00	26.82	0.00	-	-
	2	0.00	0.10	0.46	-	-	-
	1	0.00	1.82	-	-	-	-
	0	33.33	-	-	-	-	-
		0	1	2	3	4	5
		Nesting level of changed ELSE statement					

Figure 49. Enviroment of changed ELSE statements (percentages)

		Percent of DELETIONS of IF statements					
Deepest level of deleted IF statement	7	0.00	0.00	0.00	0.00	0.00	0.00
	6	0.00	0.00	0.00	0.00	0.00	0.00
	5	37.03	3.70	3.70	3.70	3.70	0.00
	4	15.69	0.00	0.00	0.00	0.00	-
	3	4.33	0.76	0.00	0.00	-	-
	2	0.79	0.54	0.00	-	-	-
	1	1.46	0.00	-	-	-	-
	0	0.00	-	-	-	-	-
		0	1	2	3	4	5
		Nesting level of deleted IF statement					

Figure 50. Enviroment of deleted IF statements (percentages)

		Percent of DELETIONS of ELSE statements					
Deepest level of deleted ELSE statement	7	0.00	0.00	0.00	0.00	0.00	0.00
	6	0.00	0.00	0.00	0.00	0.00	0.00
	5	0.00	0.00	10.53	0.00	0.00	26.32
	4	0.00	0.00	0.00	0.00	0.00	-
	3	0.00	0.00	59.09	0.00	-	-
	2	0.00	0.43	0.56	-	-	-
	1	0.00	0.00	-	-	-	-
	0	0.00	-	-	-	-	-
		0	1	2	3	4	5
		Nesting level of deleted ELSE statement					

Figure 51. Enviroment of deleted ELSE statements (percentages)

		Percent of CHANGED/DELETED IF/ELSE statements					
Deepest level of modified statement	7	0.00	0.00	0.00	0.00	0.00	0.00
	6	0.00	0.00	0.00	0.00	0.00	0.00
	5	34.78	6.52	39.13	4.35	4.35	21.74
	4	13.92	0.00	24.05	0.00	0.00	-
	3	4.89	6.85	15.50	0.00	-	-
	2	0.66	1.54	0.57	-	-	-
	1	2.19	0.30	-	-	-	-
	0	-	-	-	-	-	-
		0	1	2	3	4	5
		Nesting level of modified statement					

Figure 52. Enviroment of changed/deleted IF/ELSE statements (percentages)

AN EXPLORATORY STUDY OF
SOFTWARE DEVELOPMENT MEASURES
ACROSS COBOL PROGRAMS

by

NADINE M. VEEDER

B.S., KANSAS STATE UNIVERSITY, 1984

A MASTER'S ABSTRACT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1986

Abstract

Software managers have recognized the need for understanding as well as monitoring the software development process. This process is both complex and difficult to understand, as numerous factors affect the development process. The software development process can be better understood by objectively quantifying attributes of the software throughout the software development cycle. A measure of progress can then be obtained by applying these quantifications to a model.

This report presents the results of an exploratory study of modifications occurring to "real world" COBOL programs during the development phase. Through quantifying the types of modifications which occur throughout the software development process and the locations of these modifications, valuable insights into the software development process were obtained.

The initial objective of this analysis was to be able to discern distinct patterns from revision to revision in software development. However, due to inconsistencies in the data, this objective could not be fully realized. Rather a subset of the initial objective was accomplished. Distinctions were made between the characteristics of programs in the early phases of development and programs in the later phases of development.