

資料

COBOL プログラム輪郭作成システムの移し換えについて*

藤 村 直 美** 牛 島 和 夫**

Abstract

We have a lot of difficulty transferring software from one computer system to another. It is mainly because of the difference in computer systems as practical environments. We have implemented in various environments a COBOL program dynamic profiler, whose principal part is composed of a preprocessor and a postprocessor written in COBOL. It was first implemented on FACOM 230-45 S OS II and transferred to FACOM 230-75 Monitor 7, HITAC 8800/8700 OS 7 and IBM 370-115 DOS/VS. While transferring it, we encountered some problems, e.g. COBOL Language specification, file management, Job Control Language. This paper describes those problems and how we managed them.

1. はじめに

ある処理系で価値を認められたソフトウェアが、他の処理系に困難なく移し換えられて利用できれば、ソフトウェアの作成・使用の両面で大いに有意義である。FORTRAN や COBOL は大抵の計算機システムで利用できる。従って移し換える対象となるソフトウェアがこのような言語で記述されていれば、一応うまくいくと考えられるかもしれない。

しかしながら実際の動作環境としての計算機システムを考えると、各々異なった設計思想に基づいて作成されており、高水準言語の標準規格に陽に現われない動作環境としての差異が問題のソフトウェアの実行ないしは結果に影響を与えることがあり、これらが移し換えを難しいものになっている。我々は手続きの大部分が FORTRAN で記述された FORDAP と呼ばれる FORTRAN プログラム輪郭作成システム¹⁾をいくつかの計算機システムにおいて実現し、その時に直面した問題点等について報告した²⁾。

ここでは Fig. 1 の構成を持つシステム、すなわち COBOL プログラム輪郭作成システム (COBOLDAP

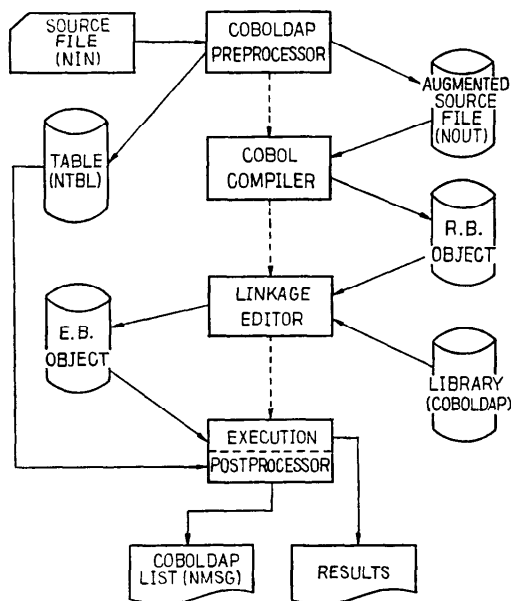


Fig. 1 General flow of COBOLDAP system.

と称する)³⁾を複数の計算機システムにおいて実現した際の問題点等について述べる。このシステムは最初九州大学情報工学科の FACOM 230-45 S OS II (以後 OS II と呼ぶ) において作成し、後に九州大学大型計算機センターの FACOM 230-75 Monitor VII

* The implementation of a COBOL program dynamic profiler in various environments by Naomi FUJIMURA and Kazuo USHIJIMA (Faculty of Engineering, Kyushu University).

** 九州大学工学部情報工学科

(以後 M7 と呼ぶ)、東京大学大型計算機センターの HITAC 8800/8700 OS 7 (以後 OS 7 と呼ぶ)、九州大学医学部付属病院医療情報室の IBM 370-115 DOS/VS (以後 DOS/VS と呼ぶ) に移し換えたものである。

この輪郭作成システムの手続きは COBOL で記述されており、FORTRAN で記述されている場合とはまた異なる問題点が現われる。すなわちハードウェアの設計、OS との関わり合い、記述言語仕様の微妙な差異、ジョブ制御文の機能といった各処理系に特有な問題に遭遇することになる。

2. COBOLDAP システム

2.1 機能とシステム構成

四つの処理系の上で実現した COBOLDAP の主たる機能は次の3点である。

- (1) COBOL プログラムの各実行文の実行回数を計数する。
- (2) 条件命令においては条件の成立した回数を計数する。
- (3) 上記の結果をわかり易く出力する。

システムの主要部は前処理部 (Fig. 1 PREPROCESSOR) と後処理部 (Fig. 1 POSTPROCESSOR) である。使用している主なファイルは次の通りである。

NIN: 計測の対象となる COBOL ソースプログラムファイルでカードや大記憶ファイル等。

NOUT: 前処理部によって実行回数計数のために必要なカウンター等が挿入された COBOL プログラムファイル。

NTBL: 実行の最後に後処理部が計測結果を出力す

るために必要な情報を含むファイル。

NMSG: 後処理部がリストを編集して出力するリストファイル。

LIBRARY: 収集した情報を実行の最後に見易く編集して出力するリスト編集ルーチン (プログラム名 COBOLDAP) が登録されている保存ファイル。

前処理部は被計測プログラムを NIN から入力し、実行回数計数のためのカウンター等を挿入して NOUT へ出力する。その変換された被計測プログラムがコンパイル後実行されることになる。計測された実行回数は後処理部によって編集・出力される (Fig. 2 参照)。Fig. 2 において EXECUTIONS 欄は各実行文の実行回数を示し、1 (AT END) は READ 命令の AT END 条件が 1 回成立したことを示す。

2.2 設計上の考慮点

COBOLDAP を設計するに当たって次のような点を考慮した。

- (1) 簡単に使用でき計測結果は理解し易いこと。
- (2) 目的プログラムにおいて計測のためのオーバーヘッドをできるだけ小さくすること。
- (3) 正常に終了しない時もそれまでに収集した情報を後処理部によって出力できること。
- (4) 他の計算機システムにできるだけ容易に移し換えられること。

我々がこのシステムを実現した四つの計算機システムは小型から超大型にまたがっているにもかかわらず、各処理系が提供している COBOL 言語はいずれも規格の高水準を満足している。従ってわざわざ手続きの記述を規格の低水準に限定してプログラムを書き

FACOM 230-455 OS2 COBOLDAP V-03 L-02 -770305-				DATE 77.05.16	TIME 11:45:02	PAGE 0001
LINE	SEQ.	A.	B.	IDENT.	EXECUTIONS	
0001		IDENTIFICATION	DIVISION.			
0002		PROGRAM-ID.	'EXAMPLE'.			1
0003						
0004						
		(中 略)				
0033		WORKING-STORAGE SECTION.				
0034		77 CR-BUF PIC X(90).				
0035						
0036		PROCEDURE	DIVISION.			1
0037		MAIN	SECTION.			1
0038		MAJIME.				1
0039		OPEN INPUT INPUT-FILE.				1
0040		OPEN OUTPUT OUTPUT-FILE.				1
0041		YOMU.				1
0042		READ INPUT-FILE INTO CR-BUF AT END GO TO OWARI.			6012	
0043		WRITE CP FROM CR-BUF.			6012	
0044		GO TO YOMU.			6011	
0045		OWARI.			6011	
0046		CLOSE INPUT-FILE.			1	
0047		CLOSE OUTPUT-FILE.			1	
0048		STOP RUN.			1	
0049		ENDCOBOL.			1	
0050						
						1 (AT END)

Fig. 2 Example of a COBOLDAP list.

難くするようなことはなかった。またそれぞれの処理系の COBOL 言語仕様には細部で微妙な差異があり、これらを同時に満足するようにプログラムを記述することも困難なので、ともかく与えられた処理系の言語仕様に従うことにした。

2.3 移し換えのための条件

Fig. 1 で示されたシステム構成を持つ COBOLDAP システムを、全体としてそのまま他の処理系へ移し換えるためには、相手の処理系が次のような条件を満足している必要がある。

- 〔I〕 COBOLDAP 前処理部が正しく動くこと。
- 〔II〕 必要なだけのファイル (Fig. 1 NIN 等) が自由に使えること。
- 〔III〕 前処理部によって作成されたファイル (Fig. 1 NOUT) が COBOL コンパイラの入力となり得ること。
- 〔IV〕 プログラムの結合・編集の段階で外部記憶から作成済みの後処理部 (Fig. 1 LIBRARY 中の COBOLDAP) を組み入れること。そのために対象となっている COBOL 言語には連絡機能が備わっていること。
- 〔V〕 実行回数計数のための命令が効率の良い機械命令に翻訳されること。
- 〔VI〕 必要なジョブ制御文が使用できること。

3. 移し換えの可能性と実現方法

移し換えの作業は 2.3 で述べた条件 (I)~(VI) の可能性を調べ実現方法を見出す所から始まる。以下個別に検討を行う。実際の移し換えに当って M7 と OS7 では FORDAP の移し換えの経験²⁾ をかなり生かすことができた。

3.1 言語仕様の相違

ここでは条件 (I) の検討を行う。使用する計算機システムが変われば ENVIRONMENT DIVISION が影響を受けるのは当然としても、前処理部・後処理部ともその部分の修正のみでは不都合な点が残る。各社の COBOL 言語仕様に細部で微妙な差があるからである。ここでは前処理部・後処理部に直接関連した問題点について述べる。主な修正箇所を Table 1 に示し、以下簡単に説明する。

- 1) M7 では計算機名は注釈として扱われるので必ずしも修正する必要はない (Table 1 第 1 項)。
- 2) OS7 では PROGRAM-ID は引用符で囲まれない。また副プログラムの入口は ENTRY で別に宣言しなくてはならない。その時入口名とプログラム名が一致してはならないので後処理部の名前を変える (Table 1 第 2, 3 項)。

Table 1 List of alterations in the preprocessor and the postprocessor.

項	変 更 箇 所	OS II	M7	OS 7	DOS/VS
1	計算機名	FACOM 230-45	FACOM 230-75	HITAC-8800	IBM-370-115
2	PROGRAM-ID. (後処理部)	'COBOLDAP'	'COBOLDAP'	AFTSVC	'COBOLDAP'
3	後処理部入口	PROCEDURE DIVISION USING 一意名	PROCEDURE DIVISION USING 一意名	ENTRY 'COBOLDAP' USING 一意名	PROCEDURE DIVISION USING 一意名
4	FILE-CONTROL NMSG NIN NOUT NTBL	U 40 U 41 U 42 U 43	SYSOUT SYSIN NOUT NTBL	UR-LP-S-SYSLST UR-CR-S-SYSLST DA-DK-S-NOUT DA-DK-S-NTBL	SYS105-UR-3203-S SYS104-UR-2501-S SYS102-DA-3340-S SYS103-DA-3340-S
5	FD (ファイル記述) (NIN)	RECORDING MODE IS F BLOCK CONTAINS 13 RECORDS RECORDS CONTAINS 60 CHARACTERS LABEL RECORD IS OMITTED.	RECORDING MODE IS F BLOCK CONTAINS 27 RECORDS LABEL RECORD IS OMITTED.	RECORDING MODE IS F LABEL RECORD IS OMITTED.	RECORDING MODE IS F LABEL RECORD IS OMITTED.
6	SPECIAL-NAMES.	無	無	C 09 IS PAGE-CHANGE.	C 01 IS PAGE-CHANGE.
7	I-O-CONTROL.	APPLY CONTROL- CHARACTER ON NMSG-FILE.	APPLY CONTROL- CHARACTER ON NMSG-FILE.	無	無
8	改ページ	WRITE record name AFTER ADVANCING 17 LINES.	WRITE record name AFTER ADVANCING 17 LINES.	WRITE record name AFTER PAGE-CHANGE.	WRITE record name AFTER PAGE-CHANGE.
9	大記憶への出力	WRITE record name [AT END 無条件命令]	WRITE record name	WRITE record name [INVALID KEY 無条件命令]	WRITE record name INVALID KEY 無条件命令
10	日 付 時 刻	ACCEPT yymmdd FROM DATE. ACCEPT hhmmssst FROM TIME.	MOVE TODAY TO yymmdd.	ACCEPT yymmdd FROM DATE. ACCEPT hhmmssst FROM TIME.	MOVE CURRENT-DATE TO mm/dd/yy. MOVE TIME-OF-DAY TO hhmmss.
11	カウンターの属性	COMP-3	COMP-2	COMP	COMP-3
12	手続きの終り	ENDCOBOL	ENDCOBOL	ソースファイルの物理的な終り	ソースファイルの物理的な終り

3) FILE-CONTROL と FD (ファイル記述) については次の 3.2 で述べる。

4) 各処理系で改ページの方法がまちまちである。OS II と M7 では Table 1 第7項のように I-O-CONTROL 段落を記述し、Table 1 第8項のように記述する。ここで 17 という値は特別な値で、ラインプリンタのコントロールチャンネル1でホームポジションスキップを行うことを示す。一方 OS7 と DOS/VS では、改ページ用のコントロールチャンネルを Table 1 第6項のように SPECIAL-NAMES 段落で記述して、第8項のような出力命令を実行する。

5) 大記憶上の順編成ファイルへの出力では、割当てられた領域を越えて出力する場合の問題がある。この条件成立時に実行する無条件命令の記述の仕方は、Table 1 第9項のように相違がある。M7 ではこれを記述してはならない。DOS/VS では記述しなくてはならない。大記憶ファイル以外への出力ではどの処理系もこの条件句は不要である。従って作業用ファイルに大記憶ファイルを使用するか否かは手続きの記述に重大な影響を及ぼす。OS II で作成した前処理部には大記憶ファイルへの出力命令が 17 カ所あるが、全て条件句を省略している。このため DOS/VS への移し換えに際してこの部分の修正を余儀なくされる。

6) 後処理部が計測結果をつけて編集した COBOL-DAP リスト (Fig. 2 参照) には、処理した日付と時刻を出力するように設計した。日付と時刻を得る手段は各処理系で異なる。特に M7 では時刻をプログラム中で参照できない。当然手続きが影響を受ける。

7) 実行回数計数のための計測時間はなるべく小さい方がよい。計数に用いるカウンターの属性の選択がこの時間に影響を及ぼす (Table 1 第11項)。詳細は 3.5 で述べる。

8) OS II と M7 には COBOL プログラムの手続きの終りを示す拡張命令 ENDCOBOL がある。そのためその直後に別のプログラム単位をつけて複数個のプログラム単位を連続してコンパイルすることが可能となる。プログラム作成に便利なのもあるが、プログラムの移換性は損われる。

9) 前処理部で使っていた一意名 TEXT が OS7 では予約語であった。こういった他の処理系の予約語を知らずに使っている場合などには、プログラム開発支援用の適当な道具があると作業が大変楽になる。ここでは 17 カ所の修正が必要であった。

結局 OS II で作成した前処理部 (約 950 枚) に対

して、M7 では修正 15 枚、挿入 0 枚、削除 4 枚、OS7 では修正 31 枚、挿入 0 枚、削除 9 枚、また DOS/VS では修正 35 枚、挿入 5 枚、削除 12 枚の変更が必要であった。これらの作業は、DOS/VS を除いて、全て TSS のテキスト修正機能を利用して行った。

なお前処理部の作成に当って各処理系において動詞を中心とした予約語の予備調査を行った^{4)~7)}。そして前処理部に必要な予約語表には、ここで述べた四つの処理系の予約語のうち必要なものはあらかじめ全て登録しておいた。従って移し換えに当ってこの部分に関連する修正は必要なかった。これらの処理系以外へ移し換える時にはこの予約語表の修正があり得る。

3.2 ファイル仕様の決定

ここでは条件 (II) の検討を行う。COBOLDAP が使用しているファイルは NIN, NOUT, NTBL, NMSG である。ここで NIN はカードか大記憶ファイル、NOUT と NTBL は操作性を考慮して大記憶の一時的ファイル、NMSG はリストファイルである (2.1 参照)。COBOL は FORTRAN に比べて扱えるファイルの種類が多い。しかし FORDAP 移し換え時の経験を踏まえて、これらのファイルは全て順編成ファイルとした。

COBOL では ENVIRONMENT DIVISION と DATA DIVISION でファイルの仕様を詳細に記述する。この記述は OS との関連で複雑な問題を含み、移し換えに重大な影響を及ぼす。いくつかのテスト・検討の後、最終的に決定したファイルの形式を Table 2 (次頁参照) に示す。これについて以下に説明する。

プログラム中のファイルと実際のファイルは FILE-CONTROL 段落中のファイル定義名を通じて実際の媒体と対応付けられる。その対応付けの仕方は各処理系ごとに異なる (Table 1 第4項)。OS II と M7 では FILE-CONTROL 段落にファイル定義名を直接記述する。例えば OS II では U 40, M7 では SYSOUT といった形である。OS7 と DOS/VS では実際のファイルの物理的な情報まで記述する。例えば OS7 では UR-LP-S-SYSLST のようにクラス-デバイス-編成-ファイル定義名となり、DOS/VS では SYS 105-UR-3203-S のようにファイル定義名-クラス-装置-編成の形になる。従って OS7 や DOS/VS では、例えば入力媒体としてカードを想定しクラスに UR (Unit Record) を指定すると、大記憶ファイルに結合することができない可能性がある。一方、OS II や M7 では媒体の物理的な特徴を記述しなくて良いこともあ

Table 2 File constants of COBOLDAP systems.

OS	ファイル名	ファイル定義名	RCDS	BLKS	BF	RECFM	LABEL
OS II	NMSG	U 40	144	1440	10	FC	O
	NIN	U 41	80	1040	13	F	O
	NOUT	U 42	80	1040	13	F	O
	NTBL	U 43	90	1170	13	F	O
M7	NMSG	SYSOUT	144	3168	22	FC	O
	NIN	SYSIN	80	2160 (1200)	27 (15)	F	O
	NOUT	NOUT	80	1200	15	F	O
	NTBL	NTBL	90	1350	15	F	O
OS7	NMSG	SYSLST	144	?	?	FC	O
	NIN	SYSIN	80	?	?	F	O
	NOUT	NOUT	80	1200	15	F	S
	NTBL	NTBL	90	1350	15	F	S
DOS/VS	NMSG	SYS 105	144	?	?	FC	O
	NIN	SYS 104	80	?	?	F	O
	NOUT	SYS 102	80	80	1	F	S
	NTBL	SYS 103	90	2970	33	F	S

RCDS: record size, BLKS: block size, BF: blocking factor, RECFM: record format, LABEL: label record, F: fixed length (blocked) record, FC: fixed length (blocked) record with control character, O: omitted, S: standard.

て、一部の媒体（例えば紙テープ）を除いて大抵の記憶媒体にそのまま結合できる。

OS II と DOS/VS では、ジョブ制御文中にレコード長やブロック長などの情報を記述することができない。そのためプログラム中で定めた形式のファイル以外へのアクセスができない。M7 と OS7 ではジョブ制御文中に FCB パラメータとしてレコード長、ブロック長等の情報を記述することができる。OS7 ではこれらの値はプログラム中のファイルの値と一致しなければならない。M7 ではブロック長は必ずしも一致しなくて良い。従ってコンパイル時に実行時の入出力バッファの大きさを指定できることと併せてジョブ制御文でブロック化率の相違を吸収することができる。いずれにしても各処理系で具体的な装置ごとに最適のブロック化率が異なるので注意が必要である。

なお一度に利用できるファイルの個数もそれぞれ異なる。OS II ではファイル定義名 U 01~U 50 に対応して 50 個、DOS/VS の BG (Background) では SYS 000~SYS 136 (一般には SYS 240 まで) で 137 個使える。OS7 と M7 では事実上制限がない。

3.3 コンパイラへの入力形式

ここでは条件(Ⅲ)の検討を行う。ここで扱っているような前処理部を持つシステムの移し換えの難関の一つは、前処理部の出力ファイルを次のコンパイラに渡す方法の問題である。M7 と OS7 では FORDAP 移し換えの経験が役立った。それでも M7 ではコンパイラに渡すファイルのブロック化率によってはコンパイ

ル時に異常終了し、必ずしも容易に成功したわけではない。

DOS/VS ではプログラムが作成したファイルをコンパイラの入力に切替えた経験がないので、予備調査とテストジョブで確認を行う必要があった。九大病院医療情報室では通常 POWER (Priority Output Writers, Execution Processors and Input Readers) のもとでジョブが処理されており、システム入力ファイル (ファイル定義名 SYSIPT) はその管理下にあるので、システム入力を切替える方法を見出すのが極めて困難であり、Fig. 3 に示す制御文でうまくいくこ

```

* ** JOB TEST
// JOB TEST
// ASSGN SYS004,X'160'
// OPTION LINK,NOLISTX,NOXREF,NO5YM
// EXEC FCBOL

** CARD COPY PROGRAM **

/*
// LBLTYP NSD(4)
// EXEC LNKEDT
// ASSGN SYS011,SYSIPT
// ASSGN SYS012,X'162'
// DLBL SYS012,'KESHITE',70/001
// EXTENT SYS012,...,4800,360
// EXEC

** COBOL PROGRAM CARD DECK **

/*
// DLBL IJSYSIN,'KESHITE'
// EXTENT SYSIPT
ASSGN SYSIPT,X'162'
// EXEC FCBOL
/*
CLOSE SYSIPT,SYSRDR
/*
* ** EOJ

```

Fig. 3 Test job for passing a created file to DOS/VS COBOL compiler.

とがわかるのに約1週間を費やしている。

3.4 プログラムの結合・編集

ここでは条件(IV)の検討を行う。COBOLDAPでは計測結果を編集して出力する副プログラム(プログラム名 COBOLDAP: Fig. 1 参照)を結合・編集時に組込んでいる。そのためこの副プログラムと被計測プログラムとの間でデータの受渡しが必要である。JIS COBOLの規格にはこうした異なるプログラム単位間の連絡機能について特に規定がないが、ここで扱っている処理系では全て拡張機能として使用できる。しかしながら実際のデータの受渡し法は各処理系で異なっている。ここではできるだけ統一的に扱うようにしているものの、3.1で述べたようにOS7では副プログラムの入口を別に宣言する必要がある。

後処理部の結合・編集はOSIIとDOS/VSでは自動組込みを、M7とOS7ではLinkage Editorに對

する補助制御文で静的に結合する方法を採用している。

3.5 カウンターの効率

ここでは条件(V)の検討を行う。すなわち実行回数計数のための加算を最小時間で行う方法を見出すことである。最初にOSIIでCOBOLDAPを作成した時は、カウンターの属性を2進数にすればそのまま2進数のload, add, storeの命令に展開され最も速くなると考えた。ところが実際に作成して使用してみると、実行回数計数のためのオーバーヘッドが大きく使用に耐えない。原因は2進数を副プログラムで内部10進数に変換して10進加算を行い、結果を再び副プログラムで2進数に変換しているためであった。主記憶上のデータの属性が2進数であるということは、必ずしも2進数の機械命令で処理されることにはならないわけである。

```
* CJOB COBOLDAP
* COBOLDAP *

** COBOL PROGRAM SOURCE DECK **

JCL and DATA DECK if needed

* JEND
```

(a) OS II

```
*NO
*USER
*COJOB
*RUN      EBNAME=E,PRECOMP,EBFLNAME=F0039,CBLDAP,LPFDFNAME=SYSOUT,
          INFDFNAME=SYSIN

** COBOL PROGRAM SOURCE DECK **

**KFILE  NOUT,FILENAME=NOUT,DISP=PASS,FCBPRM=YES,TMOD=9,RCDSIZE=80,
          BLKSIZE=1200
**KFILE  NTBL,FILENAME=NTBL,DISP=PASS,FCBPRM=YES,TMOD=9,RCDSIZE=90,
          BLKSIZE=1350
*COBOL   LISTO,SOURCE=FILE,WORK=6
**KFILE  SOURCE=FILENAME=NOUT
*LIBED
NAME     EXOTPRGM,ENTRY=ELM(EXAMPLE1)
CALL     SYSLIB,CBLLIB
SELECT   RELBIN
SELECT   COBOLDAP(COBOLDAP)
FIN
*POFILE  COBOLDAP,F0039,CBLDAP
*RUN     LPFDFNAME=SYSOUT,STEP=1

** DATA DECK **

**KFILE  NTBL,FILENAME=NTBL
JCL if needed
*JEND
```

(b) M7

```
// JOB COBOLDAP
JCL if needed
// COBOLDAP

** COBOL PROGRAM SOURCE DECK **

/*

** DATA DECK **

// END
```

(c) OS7

```
* ** JOB COBOLDAP
// JOB COBOLDAP
ASSGN SYSCLB,X'161'
// ASSGN SYSRLB,X'161'
// OPTION LINK,NOLISTX,NOXREF,NOSYM
// ASSGN SYS102,X'162'
// DLBL SYS102,'NOUT',70/001
// EXTENT SYS102,...,4400,360
// ASSGN SYS103,X'162'
// DLBL SYS103,'NTBL',70/001
// EXTENT SYS103,...,4400,360
// ASSGN SYS104,SYSLST
// ASSGN SYS105,SYSLST
// EXEC FPRECOMP
```

- ** COBOL PROGRAM SOURCE DECK **

```
/*
// ASSGN SYS004,X'160'
// DLBL IJSYSIN,'NOUT'
// EXTENT SYSIPT
- ASSGN SYSIPT,X'162'
// ASSGN SYSLSLST,IGN
// EXEC FCBOL
CLOSE SYSIPT,SYSDR
// EXEC LNKEDT
// ASSGN SYSLSLST,X'01E'
JCL if needed
// ASSGN SYS103,X'162'
// DLBL SYS103,'NTBL'
// EXTENT SYS103
// ASSGN SYS105,SYSLST
// EXEC
```

** DATA DECK **

```
/*
/*
* ** EQU
```

(d) DOS/VS

Fig. 4 Examples of JCL for COBOLDAP system.

そこで移し換えに当り、各処理系でのカウンターの属性を決定するために、各種の属性を持ったデータの加算がどのような機械命令に翻訳されるか調査を行った。その結果、4 処理系ともカウンターの属性は内部 10 進数を採用している。

3.6 ジョブ制御文のマクロ化とその移し換え

ここでは条件〔VI〕の検討を行う。COBOLDAP を実際に使用するためにはジョブ制御文を用意する必要がある。各処理系で COBOLDAP を使用するためのジョブ制御文を Fig. 4 (前頁参照) に示す。OS II と OS 7 ではジョブ制御文はマクロ化されていて利用者から詳細は見えない。使い易さという点からみてジョブ制御文のマクロ化は必須であろう⁸⁾。DOS/VS はマクロ化を検討中である。M7 ではセンターの運営方針上一般利用者がジョブ制御文のマクロ化を行うことを許していない。さらにセンター提供のマクロジョブ制御文以外使用できないため、例えば COBOLDAP を用いて計測を行うと、被計測プログラムの実行結果が消滅してしまうなどの現象が発生する。基本ジョブ制御文の使用が許されればこのような不都合は生じない。

4. 考 察

COBOLDAP は COBOL プログラムの開発・改善支援システムの一つである。このシステムの使用上の効果は主として次のようなものがある。

- (1) 作成したプログラムの動作が客観的に認識できる。
- (2) 採用したアルゴリズムの適否が実行回数から判断できる。
- (3) 実行した所とそうでない所が一目瞭然なのでプログラムの検査に有効である。
- (4) 実行回数を手掛りに手続きやデータ構造を変えるなどして、プログラムの効率改善に役立てることができる。事実 COBOLDAP 前処理部自身を COBOLDAP で計測して得られた情報をもとに表引きの方法を修正することで、処理時間を最初の版の約半分に減少させることができた。

いくつかの問題点も残されている。最初の設計ではプログラムが異常終了する時にも計測結果を出力する予定であった。しかしながら OS II 以外の処理系では異常発生時に統一的にエラー処理を行う手段が提供されておらず、現在のところ実現できないでいる。

FORDAP の移し換えでは前処理部の修正は数枚で済んでいるのに対し、COBOLDAP の移し換えでは前

処理部の修正がかなりの枚数になった (3.1 参照)。手続きの記述に際し標準規格の低水準に限定しなかったが、限定したとしても 3.1 で述べた事柄がやはり問題になると考えられる。ところで前処理部の記述言語は必ずしも COBOL である必要はない。別途 FORTRAN で記述した COBOLDAP 前処理部も稼動しており、この前処理部は数枚の修正で他の計算機システムへ移し換えることができる。この前処理部の記述にあたっては、異種言語で記述されたプログラム間でのファイルの受渡し、データの金物表現の相違等の問題が生ずる。その他記述言語の選択によるシステムの実行効率、記述の容易さ等、興味ある点も多々あるがこれらについては別の機会にゆずる。

COBOLDAP の移し換えは M7 と OS 7 は TSS で、DOS/VS はカードで作業を行った。DOS/VS への移し換えは初めてのことで、作業には約 3 週間を要した。FORDAP と COBOLDAP の移し換えを通じて、各処理系において Fig. 1 に示すような前処理部・後処理部を含むシステムを、相互に移し換える技術を確立できたと考えている。さらに別の処理系への移し換えもここであげた問題点をチェックポイントとして活かすことにより、比較的容易に行えると考えている。

DOS/VS への移し換えに際して九大病院医療情報室の柴原哲太郎氏、永田敦氏にお世話になった。ここに記して謝意を表す。

参 考 文 献

- 1) 藤村、牛島：プログラムの実行解析システムの作成と使用について、九大工学集報、Vol. 48, No. 4, pp. 95~100 (1975)。
- 2) 藤村、牛島：FORTRAN プログラム動的解析システムの移し換えについて、情報処理、Vol. 17, No. 11, pp. 1048~1055 (1976)。
- 3) 藤村、牛島：COBOL プログラム輪郭作成システムの実現と使用について、九大工学集報、Vol. 50, No. 3, pp. 209~214 (1977)。
- 4) 富士通：FACOM 230 OS II COBOL 文法編、46 SP 0714-4 (1973)。
- 5) 富士通：FACOM 230 M-V/VI/VII COBOL 文法書、75 SP-0711-2 (1974)。
- 6) 日立：OS 7 COBOL プログラミング文法、8700-3-006 (1973)。
- 7) IBM：IBM DOS/ANS COBOL 言語、N: GC 28-6394-5 (1975)。
- 8) 藤村、牛島：ジョブ制御文の移換性について、昭和 51 年度情報処理学会第 17 回全国大会予稿集 161 (1976)。

(昭和 52 年 5 月 18 日受付)