ソフトウエア工学 17－12
（1981. 2. 13）

プログラム移換作業から見たプログラミング環境の評価について

牛島和夫 ・ 田町典子
（九州大学 工学部）

1 はじめに

特定の環境下で比較的定型的な仕事を繰返していると，その環境のもつ長所は当然の存在となり有難味が薄れてくる．一方欠点に対しては本質的解決を図ることなくそれを克服するノウハウを身につけてしまい，欠点を欠点として認識しなくなるといったことが多いように思われる．そこで我々のプログラミング環境（必ずしも計算機システムに限らない）を評価するために，あえて非日常的な仕事を行いその過程を振返ってみることにした．

今回行ったのは多数のプログラムを一時に移換える作業――それぞれ異なった環境下で作成され価値を認められている５０余の汎用プログラムを我々の環境で動かそうとするもの――である．以下にその過程と考察を述べる．

2 移換作業の手順

（1）作業対象： ソースプログラムテープの形で入手した５５個の汎用プログラム（うちコンプリートプログラムは５個）を作業対象とした．これらは数値計算を中心とするFORTRANで記述されたプログラムで，カード枚数が最大4775・最小49・平均951，手続き数が最大69・最小1・平均8.9，また主プログラムを持つもの２３個，ブロックデータを持つもの５個，コモンブロックを一つでも持つもの２４個である．表１にカード枚数，ステップ数（実行文の数：ただしCONTINUE文は0，論理IF文は２と数える），コモンブロック・手続き・引数の数（なお引数の数は主プログラムとブロックデータを除いた手続きについて平均と最大を示した）等をまとめた．この表はAUDIE（後述）の結果から得られたものであり，最初からわかっていたわけではない．またデータや文書を除いた，FORTRAN部分のみに関する値であることに注意されたい．これらのプログラムに対して以下の作業を行うのに約１週間・人を要した．

（2）構成要素の分離： 各汎用プログラムのファイル中にはFORTRANのソースプログラム以外に説明文書，データ，空白行，あるいはASSEMBLY語で記述された部分等を含むものがある．これらは処理の都合上分離してそれぞれ別のファイルに移した．各構成要素はファイル中ではある程度まとまっているので，多くの場合エディタでファイルの最初と最後を調べることにより分離できる．ただしFORTRANのソースプログラム中にある非FORTRAN部分を発見するにはコンパイラの助けが必要である．実際，主プログラムとデータが交互に並んでいるプログラムファイルがあり，分離に手間がかかった．

（3）コンパイラによるチェック： エラー部分のみ出力するように指定してコンパイラにかけると，非FORTRAN部分さえなければ出力はそれほど多量にならない．対象となるプログラムは他人の使用に供するべく既に完成されたものといってよいので，作成段階のものと違って文法上の誤りは本来ないはずである．コンパイラの出力結果から次のことがわかった．

- 文法の違いによるもの（ALGORITHM文，PROGRAM文，LEVEL文，DEFINEFILE文等，あるいは実定数の指数部が３桁のもの，データ文中の定数値の繰返しを表現する書式が我々のそれと異なるもの等）．
- 環境の違いやユーザの必要への対応を考慮してユーザに何らかの操作を求めているためにそのままでは使用できないもの（配列の大きさの決定，注釈の形で入っている文の処理等）．
- ごく少数ではあるが真の誤りもある．

環境に依存する部分についてはほとんどが注釈の形で何らかの指示を与えている．しかしそれがコンパイラで検出できるとは限らないので注意が必要である．以上の点について我々の環境に合うように修正し，一応コンパイルエラーが出ない形にした．

（4）重複する手続きの発見と分離： 各汎用プログラムの中には同名の手続きを複数個含むものがあったが，プログラムを実行させるとき不便なのでこれらをそれぞれ分離した．この分離作業には最初エディタを使用していたが，ソースプログラム中での手続き間の区切り（最初と最後の行の行番号で示す）とその手続き名を得るのに手間がかかるため，その機能を持つツール（後述）を作成して役立てた．同名の手続きを複数個備えているのは次のような場合であった．

- ライブラリ（たとえばCALCOMP等）を使用できる環境用と使

（表1）　作業対象プログラムの輪郭

NO.	CARD	STEP	COMN	PROC			PARAMETERS MEAN	PARAMETERS MAX
	4775	2472	2	29	M		12.93	28
	4509	2139	2	69	M		3.87	7
	4398	1403	14	26	M	B C	2.42	7
	2754	1247	7	26	M	B	8.13	21
	2536	654	7	32	M		1.97	6
	2252	887	9	19	M	B	4.35	21
	1990	977	0	10	M	C	14.00	24
	1970	1470	1	21			10.81	16
	1794	1299	2	10	M		9.89	13
	1515	941	0	4			3.75	5
	1370	594	1	7	M	B	10.20	13
	1276	631	5	20	M	C	5.84	14
	1185	509	2	10			14.90	37
	1144	575	2	15	M		3.86	12
	1124	579	1	7	M		7.00	19
	1062	589	0	6	M		10.80	14
	747	460	6	14			7.00	10
	744	413	2	7			14.57	37
	739	417	0	7			6.14	14
	724	526	0	5	M		4.00	7
	722	450	1	9			3.56	7
	693	401	1	5	M	C	0.50	2
	685	506	2	12	M		7.09	15
	635	265	1	18	M	B	5.44	12
	612	332	0	2			9.00	12
	608	349	0	3			16.33	20
	606	324	1	5	M	C	0.50	2
	577	298	0	4	M		5.67	7
	566	339	3	9			7.33	14
	564	361	0	4			9.75	11
	556	201	0	3			16.00	25
	554	338	0	7	M		4.50	9
	552	245	0	6	M		9.20	16
	499	400	0	5			11.60	19
	431	338	0	2			9.50	11
	416	188	0	8			6.38	11
	384	281	0	2			7.50	8
	352	289	0	4			3.50	5
	349	132	0	6			6.00	7
	348	221	1	3	M		11.00	13
	345	209	0	1			16.00	16
	329	207	0	3			11.67	20
	329	134	0	3			3.67	7
	327	265	0	1			9.00	9
	299	213	0	1			13.00	13
	249	131	0	4	M		9.00	11
	236	130	0	1			6.00	6
	228	124	3	6			6.33	9
	156	94	2	1			5.00	5
	145	51	0	1			5.00	5
	141	73	0	3			5.00	6
	109	93	0	1			7.00	7
	64	42	0	1			2.00	2
	49	29	0	2			3.00	4
55	52323	26835	78	490	23	5 5	6.73	37

M : WITH MAIN PROGRAM
B : WITH BLOCK DATA
C : COMPLETE PROGRAM

用できない環境用のものが用意されている．

・ ライブラリ形式のプログラムに対し，使用例として数種のドライバ手続き（主プログラム）およびそれに伴う手続き群を含む場合．

・ ユーザの必要に応じていずれかの手続きを選択させる場合．

（5）ソースプログラム中にない手続き：　ソースプログラム中に本体のない手続きの呼出を含むプログラムでは，それがユーザの環境で使用できるかどうかを確認する必要がある．AUDIEで調べた結果次のようなものがあった．

・ 組込関数，基本外部関数．

・ ユーザが与えるべき手続き．

・ 多くの環境では使用できるライブラリ手続き．GAMMA等．

・ 特定の環境に依存する手続き．TIME等．

・ 非FORTRAN言語で記述された手続き．

（6）入出力機番とファイルの割当：　プログラムを実行させるには，入出力ファイル割当のため入出力機番を知る必要がある．機番としては定数あるいはDATA文や単純な代入文で値を与えられた変数を使用している場合が多いが，中には機番を与える変数に複雑な計算式で値を与えたり，EQUIVALENCE文を駆使したデータ構造を採用しているため説明文書に頼らざるをえないものがあった．

（7）実行：　これまでの作業で次の条件を全て満たすものは直ちに実行可能である．

・ 主プログラムがある．

・ 入力がない．または入力があってそれに対応するデータがあり，入力機番がわかっている．

・ 出力がない．または出力があって出力機番および出力ファイルの形式がわかっている．

・ コンパイルエラーがない．

・ 必要な手続きがすべて使用できる．

今のところソースプログラムそのものや注釈の形で与えられた文書を丁寧に読む作業が完全に終っていないので，必要なデータや手続きが陽に記述されていないもの，修正がそれほど容易ではない機種依存部分を含むものは実行可能プログラムに含めていない．反面実行可能としたプログラムに陽に発見されていないエラー（必要な文が注釈のまま取り残されているなど）が含まれている可能性もある．

3　汎用プログラムの移換性

多くの環境への移換性（移換の容易さ）を考慮してFORTRANでプログラムを書く場合の考慮点については既にいくつかの研究がある［4，5，6，7，8］．今回そのようにして書かれた（と思われる）プログラムを新しい環境に移換える作業を行ったが，この作業を通じて気付いた点を付加える．最大の問題は汎用プログラムの満たすべき条件が確立していないことである．個々のプログラムは移換性にかな

りの考慮を払っていることがうかがえるが，それぞれが異なった手法でそれを実現しているために，一時にたくさんのプログラムを移換えようとすると個々のプログラムの特徴をとらえるのに手間がかかる．文書のあり方・ソースファイル構成・環境依存部分への対応・テスト方法等についての何らかの標準が定められ普及することが求められるが，そのような標準がない以上，それについて記述した移換のための文書といったものが添えられていることが望ましい．

（1）ソースファイルの構成：　ソースファイルの構成を述べる文書をファイル中にではなく添付文書として人が読める形にしておけば，これらを同時に維持管理する問題は生ずるものの作業は格段に楽になる．ファイル中で文書・ソースプログラム・データ等がどんな順序ではいっているか，それらの区切りはどこかを調べるのは意外に大変だからである．複数の言語が使用されている場合，複数組のデータがある場合，あるいは必要に応じて使い分けなければならない複数個の手続きがある場合，複数個のドライバ手続きそれぞれにデータがある場合などもあって，特にファイルが大きい場合にはかなり手間がかかる．

（2）環境依存部分：　環境依存部分については多くの場合注釈で適切な指示をしている．しかしその手法は統一されておらず，機種に応じて指定された注釈行の先頭の“Ｃ”を“　”に置換えることにより目的を達せられるものもある半面，必要な文や手続きの書き方あるいは機能のみが文章で記述されているものもあり，その記述方法もまちまちである．

（3）その他の文書：　ユーザのための使用手引書は最低限必要である．特にデータの書式や入出力ファイルの割当方法，ライブラリプログラムに対しては実引数並びの意味を含むその呼出方法は不可欠である．また保守やレベルアップに役立てるために，手続き間の呼出構造や手続き間のデータ通信機構，あるいはアルゴリズムや変数の意味等を含む文書もあった方がよい．

（4）テストデータ：　実行可能なプログラムのなかには複数個のドライバ手続きやデータによって複数のテストができるものがいくつかあったが，それらのテストを全部行ってみても一度も実行されない文や一度も呼出されない手続きが多く，ソフトウェアテストの立場からみると必ずしも十分なテストデータが与えられているとはいえない．これらの汎用プログラムはテスト済みであるから，ドライバ手続きやテストデータはユーザに典型的な使用例を示すために与えられているのだろう．一方これらの汎用プログラムはそれぞれ別の作成者によりそれぞれ別の環境で作成されたものであり，新しい環境でそのまま無修正で動くものばかりとは限らない．また精度が問題となるかも知れない．そのためのテスト例をどう付けるかの問題が残されている．

4　ツールの評価

（1）エディタ：　この作業に使用したのはFACOM OSIV/F4下のエディタであり，リスト・挿入・削除・置換等の機能，特定の文字列にパタンマッチする行全てをリストする機能がある．この他にコマンドのマクロ化機能や強力なパタンマッチ機能（正規表現が使用できるなど）があればこの作業はもっと容易であっただろう．一方汎用エディタの限界として，範囲指定を行番号あるいはそれに準ずるものによってしかできないこと，名前によるパタンマッチができないことが不便であり，FORTRANでは空白が意味を持たないことも障害となる．

（2）コンパイラとリンケージエディタ：　手続き内の文法エラーについてはコンパイラが最も有力なツールである．手続き間関係にかかわるエラーについてはリンケージエディタに頼らざるをえないが，実

```
-UNLIST
*       LIST THE LINE NUMBER OF EACH FORTRAN PROGRAM UNIT
*
*       AUTHOR.         DR. N. FUJIMURA.
*       DATE-WRITTEN. 80/10/03.
*
        LINENO = '0'
        LINE   = '**MAIN**'
        STARTLINE = '1'
        EJECT('SYSPOT')
        SYSPOT =
        SYSPOT = '   FROM     TO  PROGRAM NAME'
        SYSPOT =
*
INPUT   BUF = SYSPIT                            /F(END)
        LINENO = LINENO + '1'
        BUF *BUF/'72'*
        BUF ANCHOR() 'C'                        /S(INPUT)
        BUF ANCHOR() */'5'* ' '                 /S(ST)
        BUF ANCHOR() */'5'* '0'                 /S(ST)F(INPUT)
ST      BUF ' FORMAT'                           /S(INPUT)
        BUF '    END    '                       /S(OUTPUT)
        BUF ' SUBROUTINE'                       /S(PROG)
        BUF 'FUNCTION'                          /S(PROG)
        BUF ' BLOCK '                           /S(PROG)F(INPUT)
*
PROG    LINE = BUF
DLTBLK  LINE ANCHOR() ' ' =                     /S(DLTBLK)F(INPUT)
*
OUTPUT SYSPOT = ' ' DUPL(' ','6' - SIZE(STARTLINE)) STARTLINE
.               ' ' DUPL(' ','6' - SIZE(LINENO))    LINENO
.               '  ' LINE
        STARTLINE = LINENO + '1'
        LINE = '**MAIN**'                       /(INPUT) ;
*
END
```

（図１）　手続き名列挙ツール（SNOBOL3 による）

引数と仮引数の対応がとれていないことが実行時にならないとわからないなど，リンケージエディタのチェックでも十分とはいえない．

（3）ＡＵＤＩＥ：　手続き間情報を解析し，表形式で出力する自作のツールである［1，2］．ソースプログラム中にない手続きを含めて，手続きの呼出関係の把握や実行可能なプログラムのテスト結果のまとめに役立った．また表１の統計情報もAUDIEの出力をもとに作成した．一方AUDIEの制限として，同名の手続き（主プログラム，ブロックデータを含む）が複数個あると最初のものしか見ないため，あらかじめ同名手続きを分離しておかねばならない不便があった．また１０００ステップを越える大きな手続きがあるとその手続きが解析対象から外されるが，今回はそのような例には一件も出会わなかった．

（4）ＳＮＯＢＯＬ：SNOBOLでは挿入・削除を伴うパタンマッチを一文で書くことができ，その組合せにより文字列に関するかなり複雑な操作も簡単に記述できる［3］．これを用いてエディタの機能を補うツールを容易に作成することができ，今回の作業では手続きの区切りと手続き名とを出力するツールを作成した（図１参照）．

5　おわりに

他人の作成したプログラムをソーステキストレベルで扱うためにはいろいろなツールが必要となる．プログラムがかなり大きい場合にツールなしで作業を行うことは非常に困難であり，必要な情報をよりコンパクトでわかりやすい形で得ることが重要となる．我々の作成したAUDIEは静的解析情報・動的解析情報を手続きと大域データに着目して整理し出力する．一方，多くのプログラムを扱う場合にはちょっとしたツールの有無や使い勝手が作業能率に大きく影響する．そのような場合ユーザが自分の望む機能を（SNOBOLなどを用いて）容易に実現できるという環境［3］は，なんでもやってくれる大型ツールがあるのとまた別の意味で有益である．

謝辞　移換作業について当研究室の藤村直美助手より貴重な示唆をいただいた．ここに謝意を表したい．なお本稿は九大大型計算機センターのＪＥＦシステムを用いて作成した．

参考文献

［1］牛島・田町：実行回数を含む手続き間情報の解析と整理のツール，電子通信学会技術研究報告，VOL.EC79-57，1980．

［2］牛島・田町：手続き間情報の解析と整理のツールAUDIEの使用経験，第２２回プログラミングシンポジウム報告集，PP.92-99，1981．

［3］木村：SOFTWARE TOOLとは何か，情報処理，VOL.20，NO.8，PP.673-680，1979．

［4］藤田・葛山・川原：プログラムの移植について，情報処理，VOL.21，NO.11，PP.1128-1135，1980．

［5］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．

［6］T.J.AIRD，E.L.BATTISTE，AND W.C.GREGORY: PORTABILITY OF MATHEMATICAL SOFTWARE CODED IN FORTRAN，ACM TOMS，VOL.3，NO.2.PP.113-127，1977．

［7］P.A.FOX，A.D.HALL，AND N.L.SCHRYER:THE PORT MATHEMATICAL SUBROUTINE LIBRARY，ACM TOMS，VOL.4，NO.2，PP.104-126，1978．

［8］A.S.TANENBAUM，P.KLINT，AND W.BOHN: GUIDELINE FOR SOFTWARE PORTABILITY，SOFTWARE-PRACTICE AND EXPERIENCE，VOL.8，NO.6，PP.681-698，1978．