CREATING BANNER PAGES

Edmund C. Arranga
Business Programmer/Analyst
McDonnell Douglas Corp.
3855 Lakewood Boulevard
Long Beach, CA 90846

INTRODUCTION

This paper describes for the HP programmer a process to create banner pages for reports. The banner pages will contain report distribution information specific to the reports, yet remain independent of the reports' creation. The process to create report specific banner pages with mailing information is divided into 3 major steps or components. These components are MPE commands, SPOOK, and COBOL. Each component in the process will be examined and explained individually before the entire process is presented.

I.) MPE COMMANDS (Multiprogramming Executive operating system)

The banner page creation process uses 6 MPE commands. They are: BUILD, FILE, HEADOFF, HEADON, OUTFENCE, and SHOWOUT. For reference, the exact syntax used by the 6 MPE commands is given below.

```
1.) :BUILD SO;REC=-80,,F,ASCII;DISC=100 * :BUILD SI;REC=-80,,F,ASCII;DISC=10
2.) :FILE BANNER;DEV=137,  :FILE BPAGE;REC=-133,,F,ASCII;DEV=137,1,1
3.) :HEADOFF 137
4.) :HEADON 137
5.) :OUTFENCE 12;LDEV=137
6.) :SHOWOUT JOB=@:DEV=137
```

SHOWOUT

The MPE SHOWOUT command reports the status of output device files. These are the files or reports waiting to print. To illustrate, the SHOWOUT command is demonstrated below.

```
:SHOWOUT JOB=@;DEV=137

DEV/CL    DFID      JOBNUM  FNAME    STATE FRM SPACE RANK PRI #C
137       #074      #J'133  MAILINFO READY          40     D 1   1
137       #075      #J'294  REPORT1  READY         252     D 1   1
```

*The HP 3000 prompts are used to indicate specific levels of user interaction.
The colon (:) indicates the operating system level
The greater than sign (>) indicates an interactive SPOOK session
Example MPE commands differ slightly from the syntax of the above commands

```
2 FILES (DISPLAYED):
   0 ACTIVE
   2 READY; INCLUDING 2 SPOOFLES,  2 DEFERRED
   0 OPENED; INCLUDING 0 SPOOFLES
   0 LOCKED; INCLUDING 0 SPOOFLES
   2 SPOOFLES:   272 SECTORS
OUTFENCE = 1
```

In this example two output spool files, MAILINFO and REPORT1 are waiting to print on device 137. Their DFID's (device file ID) are #074, and #075 respectively. The FNAME and DFID will be used to uniquely identify output spool files to tag with report distribution information.

For demonstration purposes it is assumed that MAILINFO contains report distribution information specific to REPORT1. If the two files are joined the result would be a report with a banner that contained distribution information. The ability to join these two files is examined next in the section on SPOOK.

II.) SPOOK

The second component in the banner creation process uses SPOOK. SPOOK.PUB.SYS is a utility program supplied with MPE which allows the user to interrogate and manipulate spool files. Different versions of MPE run different versions of SPOOK. Nonetheless, the command sets are identical for SPOOK, SPOOK4 and SPOOK5. This paper will use the term SPOOK as a generic reference to all three versions.

SPOOK's APPEND COMMAND

The SPOOK APPEND command, "APPENDS PART OR ALL OF AN OUTPUT SPOOL FILE TO ANOTHER OUTPUT SPOOL FILE." Herein lies the key to creating banner pages. A report waiting to print (an output spool file) can be appended to another report (another output spool file) waiting to print.

Before the APPEND command is demonstrated SPOOK has two other useful capabilities which will be briefly discussed. The first is, the ability to use MPE commands while running SPOOK. In fact, any MPE command that can be accessed programmatically by the COMMAND intrinsic can be accessed by SPOOK. The second, is the ability to back reference a file equation in the APPEND command.

These capabilities along with the APPEND command are illustrated below.

```
:RUN SPOOK.PUB.SYS

SPOOK G.02.BO  (C) HEWLETT-PACKARD CO., 1983
>FILE BANNER;DEV=137,12,1
>APPEND #074,#075;ALL,*BANNER
>APPEND END
```

```
>SHOWOUT JOB=@;DEV=137

DEV/CL    DFID     JOBNUM  FNAME    STATE FRM SPACE RANK PRI #C
137       #076     #S1076  BANNER   ACTIVE       272    1 12  1
137       #074     #J'133  MAILINFO READY         40    D 1   1
137       #075     #J'294  REPORT1  READY        252    D 1   1


3 FILES (DISPLAYED):
   1 ACTIVE
   3 READY; INCLUDING 3 SPOOFLES,  2 DEFERRED
   0 OPENED; INCLUDING 0 SPOOFLES
   0 LOCKED; INCLUDING 0 SPOOFLES
   3 SPOOFLES:   564 SECTORS
OUTFENCE = 1
```

MAILINFO and REPORT1 have been appended to create a new spool file, BANNER with the parameters assigned in the file equation. BANNER is 'ACTIVE' (printing) on device 137 because of the output priority assigned in the file equation.

<u>REDIRECTING SPOOK's STDLIST and STDIN</u>

Until now the information returned by the SHOWOUT command has been directed to $STDLIST, or the terminal. To capture the same output spool file information in a more permanent manner, STDLIST can be redirected to a disc file. First the BUILD command is used to construct a MPE file.

```
:BUILD SO;REC=-80,,F,ASCII;DISC=100
```

This permanent disc file, SO is where SPOOK will be directed to return information.

```
:RUN SPOOK.PUB.SYS;STDLIST=SO
```

MPE now runs SPOOK but the terminal does not display the familiar SPOOK signon. Instead this information has been sent to the file SO. Continuing, the SHOWOUT command is entered;

```
SHOWOUT JOB=@;DEV=137     (again SO captures the information)
EXIT                      (terminate SPOOK)
:                         (return to the operating system)
```

At this point, the ability to redirect STDLIST, and the ability to use MPE commands within SPOOK has allowed the programmer to capture output spool file information on a permanent disc file.

In the same manner SPOOK is directed to send information to a disc file, SPOOK can be directed to receive commands from a disc file. In a previous example the SPOOK APPEND command was used to create a new spool file. The same commands used then interactively, now can be directed to SPOOK from a disc file. The disc file, SI contains the information as follows:

```
 FILE BANNER;DEV=137,12,1
 APPEND #074, #075;ALL,*BANNER
 APPEND END
 EXIT
```

SPOOK is run now with STDIN equated to SI, the command file.

```
:RUN SPOOK.PUB.SYS;STDIN=SI
```

SPOOK signs on, looks to SI for its commands, carries out the commands and signs off, per the EXIT command.

CALLING SPOOK PROGRAMMATICALLY

This section introduces a small SPL (systems programming language) program that calls SPOOK. (NOTE - SPL is not mandatory to create banner pages. All the commands used in the SPL program can be coded using a higher level language such as Pascal or COBOL. The choice is strictly up to the programmer. By using SPL however, the programmer avoids the wordiness of COBOL and has available a small utility program useful in other applications).

The program listed below named 'CALLSPK' equates STDIN to the command file SI. A COBOL program introduced in the next section will be responsible to build SI with the appropriate APPEND and SHOWOUT commands. The file SO is equated to STDLIST and is used to capture the information generated by the SHOWOUT command.

```
  $CONTROL SUBPROGRAM
  BEGIN
  PROCEDURE CALLSPK;
  BEGIN
  INTRINSIC CREATEPROCESS, ACTIVATE, KILL;
  OWN ARRAY B (0:2);
  OWN INTEGER ARRAY IA (0:2) :=1(9,8,0);
  OWN LOGICAL ARRAY LA (0:1);
  OWN BYTE ARRAY BA1 (0:39) := 1("SO.PUB.PRODW",%15);   equate STDLIST to SO
  OWN BYTE ARRAY BA2 (0:39) := 1("SI.PUB.PRODW',%15);   equate STDIN to SI
  OWN BYTE ARRAY PRINTPROC (0:39):= "SPOOK.PUB.SYS";    call SPOOK
  INTEGER ERROR,PIN;
  LA(0) := @BA1;
  LA(1) := @BA2;
  CREATEPROCESS (ERROR,PIN,PRINTPROC,IA,LA);
  ACTIVATE (PIN,2);
  KILL (PIN);
  END;
  END.
```

(III.) COBOL

The third and final component in the banner creation process uses a COBOL program called 'MAKEBANN'. 'MAKEBANN' is responsible to create the banner page specific to the output spool file waiting to print and to construct the command file SI. SI is in turn used by SPOOK to append the banner page to the spool file.

THE BANNER PAGE

Below is a sample of the banner page without the mailing information. 'MAKEBANN' will be responsible to supply this information specific to each report.

```
                 SEND REPORT TO:

                 NAME:
                 DEPT:
            MAIL CODE:


     FOR REPORT DISTRIBUTION UPDATES PLEASE CALL THE PROGRAMMING DEPT.
     OR UPDATE THIS PAGE AND MAIL TO THE PROGRAMMING DEPT.
```

Until now the output spool file MAILINFO was assumed to contain the mailing information for REPORT1. This assumption is no longer needed. Instead a file called 'USERINFO' will be used which contains the mailing information needed by the banner page. For this example 'USERINFO' is a flat MPE file, although it might well be a small data base or KSAM file. Regardless of the structure, the COBOL program will match the FNAME returned by the SHOWOUT command to a FNAME in 'USERINFO'. Here is where unique FNAMEs become important. For installations without unique FNAMEs the job stream of that particular job can be changed to equate either default FNAMEs (i.e. COBLIST, ASKLIST etc.) or conflicting FNAMEs to unique FNAMEs.

The layout and the information of the file 'USERINFO' is as follows:

FNAME	LSTNM	FSTNM	DEPT	MAILCD
REPORT1	RAINS	PAT	P81	204-36
REPORT3	PUBLIC	JOHN	W373	LOC-3A
REPORT4	HENRY	FRED	W454	LOC-21

'USERINFO' remains as a permanent disc file on the system and is updated periodically to include new jobs or to change existing information.

If the SHOWOUT command returns the following information:

```
DEV/CL   DFID    JOBNUM  FNAME    STATE FRM SPACE RANK PRI #C
137      #02219  #S580   REPORT1  READY          32    1  8   1
137      #02220  #S580   REPORT2  READY          32    2  8   1
137      #02221  #S580   REPORT3  READY          32    3  8   1
```

then 'MAKEBANN' will produce banner pages with mailing information for REPORT1 and REPORT3.

As an added feature the programmer might at this point want to send reports to multiple printers. With a litle extra work 'USERINFO' could contain several printer destinations. The COBOL program would check this field and make appropriate additions to the command file SI which is passed to SPOOK. For example if REPORT2 is to be printed at device 137 and at device 136 the command file SI would contain the following:

```
FILE BANNER;DEV=137,12,1
APPEND #DFID,#DFID;ALL,*BANNER
APPEND END
FILE BANNER;DEV=136,12,1
APPEND #DFID,#DFID;ALL,*BANNER
APPEND END
PURGE #DFID, #DFID
EXIT
```

THE PROCESS

The entire process runs as a background job. Once a minute the program wakes up to check the output spool files. These output spool files are compared to the FNAMEs of the 'USERINFO' file now stored in a table in 'MAKEBANN'. If a match occurs a banner page is generated with the mailing information. 'MAKEBANN' again checks the output spool files, this time to return the #DFID of the newly created banner page (BPAGE). The command file is then loaded and executed with the commands to append the banner page to the proper report. This process then repeats itself every minute.

SUMMARY

Reports can now be generated that facilitate distribution. The HP 3000 has readily available the tools to accomplish this task. The HP programmer no longer has to rely on the standard sysout page to route reports. Instead banner pages can now be created as they are needed for existing and future reports.

APPENDIX A
SOURCE LISTING FOR ALL PROGRAMS
PROGRAM: 'MAKEBANN'

```
$CONTROL NOLIST
IDENTIFICATION DIVISION.
PROGRAM-ID.  CREATE-BANNER.
*AUTHOR.  ECA.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. HP-3000.
OBJECT-COMPUTER. HP-3000.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
 SELECT 100-USERINFO-FILE        ASSIGN  'USERINFO'.
 SELECT 150-SPOOL-INFO-FILE      ASSIGN  'SO'.
 SELECT 200-SPOOK-COMMAND-FILE   ASSIGN  'SI'.
 SELECT 250-BANNER-PAGE-FILE     ASSIGN  'BPAGE'.
DATA DIVISION.
FILE SECTION.

FD  100-USERINFO-FILE
    RECORD CONTAINS 80 CHARACTERS.
01  100-USERINFO-RECORDS     PIC X(80).

FD  150-SPOOL-INFO-FILE
    LABEL RECORDS ARE STANDARD
    RECORD CONTAINS 80 CHARACTERS
    RECORDING MODE IS FIXED.
01  150-SPOOL-INFO-RECORD    PIC X(80).

FD  200-SPOOK-COMMAND-FILE
    LABEL RECORDS ARE STANDARD
    RECORD CONTAINS 80 CHARACTERS
    RECORDING MODE IS FIXED.
01  200-SPOOK-COMMAND-RECORD PIC X(80).

FD  250-BANNER-PAGE-FILE
    LABEL RECORDS ARE STANDARD
    RECORDS CONTAINS 133 CHARACTERS
    RECORDING MODE IS FIXED.
01  250-BANNER-PAGE-RECORD    PIC X(133).


WORKING-STORAGE SECTION.

01  200-COMMAND-LINE-1.
    05  FILLER             PIC X(80) VALUE SPACES.

01  200-COMMAND-LINE-2.
    05  FILLER             PIC X(25) VALUE.
        'FILE BANNER;DEV=137,13,'.
    05  200-COPIES         PIC X(04) VALUE SPACES.
```

```
01  200-COMMAND-LINE-3.
    05  FILLER              PIC X(07) VALUE 'APPEND '.
    05  200-DFID-BANNER     PIC X(06) VALUE SPACES.
    05  FILLER              PIC X(01) VALUE ','.
    05  200-DFID-REPORT     PIC X(06) VALUE SPACES.
    FILLER                  PIC X(12) VALUE
    ';ALL,*BANNER'.

01  200-COMMAND-LINE-4.
    05  FILLER              PIC X(10) VALUE 'APPEND END'.

01  200-COMMAND-LINE-5.
    05  FILLER                  PIC X(06) VALUE 'PURGE '.
    05  200-P-DFID-REPORT       PIC X(06) VALUE SPACES.
    05  FILLER                  PIC X(01) VALUE ','.
    05  200-P-DFID-BANNER       PIC X(06) VALUE SPACES.

01  250-BANNER-LINE-1.
    05  FILLER      PIC X(20) VALUE SPACES.
    05  FILLER      PIC X(16) VALUE 'SEND REPORT TO: '.
    05  FILLER      PIC X(97) VALUE SPACES.

01  250-BANNER-LINE-2.
    05  FILLER      PIC X(20) VALUE SPACES.
    05  FILLER      PIC X(06) VALUE 'NAME: '.
    05  250-B-LSTNM PIC X(20) VALUE SPACES.
    05  250-B-FSTNM PIC X(12) VALUE SPACES.
    05  FILLER      PIC X(75) VALUE SPACES.

01  250-BANNER-LINE-3.
    05  FILLER      PIC X(20) VALUE SPACES.
    05  FILLER      PIC X(06) VALUE 'DEPT: '.
    05  250-B-DEPT  PIC X(04) VALUE SPACES.
    05  FILLER      PIC X(103) VALUE SPACES.

01  250-BANNER-LINE-4.
    05  FILLER      PIC X(15) VALUE SPACES.
    05  FILLER      PIC X(11) VALUE 'MAIL CODE: '.
    05  250-B-DEPT  PIC X(06) VALUE SPACES.
    05  FILLER      PIC X(98) VALUE SPACES.

01  250-BANNER-LINE-5.
    05  FILLER      PIC X(10) VALUE SPACES.
    05  FILLER      PIC X(32) VALUE
    'FOR REPORT DISTRIBUTION UPDATES '.
    05  FILLER      PIC X(23) VALUE
    'PLEASE CALL PROGRAMMING'.
    05  FILLER      PIC X(68) VALUE SPACES.

01  250-BANNER-LINE-6.
    05  FILLER      PIC X(10) VALUE SPACES.
    05  FILLER      PIC X(29) VALUE
    'OR UPDATE THIS PAGE AND MAIL '.
    05  FILLER      PIC X(25) VALUE
    'TO THE PROGRAMMING DEPT.'.
    05  FILLER      PIC X(69) VALUE SPACES.
```

```
01  300-USERINFO-RECORD.
    05  300-FNAME           PIC X(08).
    05  FILLER              PIC X(01).
    05  300-LSTNM           PIC X(20).
    05  FILLER              PIC X(01).
    05  300-LSTNM           PIC X(12).
    05  FILLER              PIC X(01).
    05  300-DEPT            PIC X(04).
    05  FILLER              PIC X(01).
    05  300-MAILCD          PIC X(06).
    05  FILLER              PIC X(01).
    05  300-COPIES          PIC X(02).

01  310-SPOOK-SPOOL-RECORD.
    05  310-DEVICE          PIC X(08).
    05  FILLER              PIC X(01).
    05  310-DFID            PIC X(06).
    05  FILLER              PIC X(02).
    05  310-JOBNUM          PIC X(06).
    05  FILLER              PIC X(02).
    05  310-FNAME           PIC X(08).
    05  FILLER              PIC X(01).
    05  310-STATE           PIC X(06).
    05  FILLER              PIC X(06).
    05  310-SPACE           PIC X(06).
    05  FILLER              PIC X(04).
    05  310-RANK            PIC X(01).
    05  310-PRI             PIC X(02).
    05  310-COPIES          PIC X(04).

01  400-SWITCHES-AND-THINGS.
    05  USER-INFO-COUNT   PIC 9(02) VALUE 0.
    05  EOF-USER-INFO     PIC X(01) VALUE 'N'.
    05  EOF-SP            PIC X(01) VALUE 'N'.
    05  EOF-BPAGE         PIC X(01) VALUE 'N'.
    05  FIND-BPAGE        PIC X(01) VALUE 'N''
    05  FIND-USER         PIC X(01) VALUE 'N'.
    05  DFID-BANNER       PIC X(06) VALUE SPACES.
    05  DFID-REPORT       PIC X(06) VALUE SPACES.
    05  COPIES            PIC X(04) VALUE SPACES.
    05  TABLE-SEARCH      PIC X(01) VALUE 'N'.
    05  END-OF-TIME       PIC X(01) VALUE 'N'.

01  500-USER-INFO-TABLE.
    05  500-USER-INFO OCCURS 1 TO 100 TIMES
                      DEPENDING ON USER-INFO-COUNT
                      ASCENDING KEY IS 500-FNAME
                      INDEXED BY USER-IDX.
    10  500-FNAME               PIC X(08).
    10  FILLER                  PIC X(01).
    10  500-LSTNM               PIC X(20).
    10  FILLER                  PIC X(01).
    10  500-FSTNM               PIC X(12).
    10  FILLER                  PIC X(01).
    10  500-DEPT                PIC X(04).
    10  FILLER                  PIC X(01).
    10  500-MAILCD              PIC X(06).
    10  FILLER                  PIC X(01).
    10  500 COPIES              PIC X(02).


PROCEDURE DIVISION.

A000-MAIN.
     OPEN INPUT 100-USERINFO-FILE.
     SET USER-IDX TO 1.
     READ 100-USERINFO-FILE INTO 300-USERINFO-RECORD
        AT END MOVE 'Y' TO EOF-USER-INFO.
     PERFORM A020-LOAD-USER-INFO-TABLE UNTIL
                         EOF-USER-INFO = 'Y'.
     CLOSE 100-USERINFO-FILE.
     PERFORM A035 PROCESS UNTIL END-OF-TIME = 'Y'.

A035-PROCESS.
     CALL 'WAIT'.
     PERFORM A030-LOAD-SHOWOUT-INFO.
     CALL 'CALLSPK'.

     MOVE 'N' TO TABLE-SEARCH.
     MOVE 'N' TO FIND-USER.
     MOVE 'N' TO EOF-SP.
     SET USER-IDX TO 1.
     OPEN I-O   150-SPOOL-INFO-FILE.
     PERFORM A040-READ-SPOOL-INFO-FILE
           UNTIL EOF-SP = 'Y' OR FIND-USER = 'Y'.
     CLOSE 150-SPOOL-INFO-FILE.
     IF FIND-USER = 'Y'
     THEN
        PERFORM A050-CREATE-BANNER-PAGE
     ELSE
        GO TO A035-PROCESS.

     CALL 'CALLSPK'.

     MOVE 'N' TO EOF-BPAGE.
     MOVE 'N' TO FIND-BPAGE.
     OPEN I-O 150-SPOOL-INFO-FILE.
     READ 150-SPOOL-INFO-FILE INTO
        310-SPOOK-SPOOL-RECORD AT END
         MOVE 'Y' TO EOF-BPAGE.
     PERFORM A060-FIND-DFID-BPAGE UNTIL
       EOF-BPAGE = 'Y' OR FIND-BPAGE = 'Y'.
     CLOSE 150-SPOOL-INFO-FILE.

     PERFORM A070-PREPARE-APPEND-COMMANDS.

     CALL 'CALLSPK'.

     GO TO A035-PROCESS.
```

```
A020-LOAD-USER-INFO-TABLE.
     ADD 1 TO USER-INFO-COUNT.
     MOVE 300-FNAME  TO 500-FNAME (USER-INFO-COUNT).
     MOVE 300-LSTNM  TO 500-LSTNM (USER-INFO-COUNT).
     MOVE 300-FSTNM  TO 500-FSTNM (USER-INFO-COUNT).
     MOVE 300-DEPT   TO 500-DEPT (USER-INFO-COUNT).
     MOVE 300-MAILCD TO 500-MAILCD (USER-INFO-COUNT).
     MOVE 300-COPIES TO 500-COPIES (USER-INFO-COUNT).

     SET USER-IDX UP BY 1.
     READ 100-USERINFO-FILE INTO 300-USERINFO-RECORD
        AT END MOVE 'Y' TO EOF-USER-INFO.

A030-LOAD-SHOWOUT-INFO.
     OPEN I-O 200-SPOOK-COMMAND-FILE.
     MOVE 'SHOWOUT JOB=@;DEV=LP226' TO
          200-COMMAND-LINE-1.
     WRITE 200-SPOOK-COMMAND-RECORD FROM
           200-COMMAND-LINE-1.
     MOVE 'EXIT' TO 200-COMMAND-LINE-1.
     WRITE 200-SPOOK-COMMAND-RECORD FROM
           200-COMMAND-LINE-1.
     CLOSE 200-SPOOK-COMMAND-FILE.

A040-READ-SPOOL-INFO-FILE.
     READ 150-SPOOL-INFO-FILE INTO
          310-SPOOL-RECORD AT END
          MOVE 'Y' TO EOF-SP.
     IF 310-FNAME = 'STOPSPK'
     THEN
        PERFORM A080-END.
     IF 310-STATE = 'READY'
     THEN
      SEARCH ALL 500-USER-INFO
       AT END MOVE 'Y' TO TABLE-SEARCH
        WHEN 500-FNAME (USER-IDX) = 310-FNAME
         MOVE 310-DFID TO DFID-REPORT
         MOVE 500-LSTNM (USER-IDX)  TO 250-B-LSTNM
         MOVE 500-FSTNM (USER-IDX)  TO 250-B-FSTNM
         MOVE 500-DEPT (USER-IDX)   TO 250-B-DEPT
         MOVE 500-MAILCD (USER-IDX) TO 250-B-CODE
         MOVE 310-COPIES TO COPIES
         MOVE 'Y' TO FIND-USER
         MOVE 'Y' TO EOF-SP.

A050-CREATE-BANNER-PAGE.
     OPEN OUTPUT 250-BANNER-PAGE-FILE
     WRITE 250-BANNER-PAGE-RECORD FROM
           250-BANNER-LINE-1 AFTER 10.
     WRITE 250-BANNER-PAGE-RECORD FROM
           250-BANNER-LINE-2 AFTER 2.
     WRITE 250-BANNER-PAGE-RECORD FROM
           250-BANNER-LINE-3.
     WRITE 250-BANNER-PAGE-RECORD FROM
     WRITE 250-BANNER-PAGE-RECORD FROM
           250-BANNER-LINE-4.
     WRITE 250-BANNER-PAGE-RECORD FROM
           250-BANNER-LINE-5 AFTER 5.
     WRITE 250-BANNER-PAGE-RECORD FROM
           250-BANNER-LINE-6.
     CLOSE 250-BANNER-PAGE-FILE.

A060-FIND-DFID-BPAGE.
     IF 310-FNAME = 'BPAGE'
     THEN
        MOVE 310-DFID TO DFID-BANNER
         MOVE 'Y' TO FIND-BPAGE.

      IF FIND-BPAGE = 'N'
      THEN
      READ 150-SPOOL-INFO-FILE INTO
           310-SPOOK-SPOOL-RECORD AT END
            MOVE 'Y' TO EOF-BPAGE.

A070-PREPARE-APPEND-COMMANDS.
     OPEN I-O 200-SPOOK-COMMAND-FILE.
     MOVE COPIES      TO 200-COPIES
     MOVE DFID-BANNER TO 200-DFID-BANNER,
                         200-P-DFID-BANNER.
     MOVE DFID-REPORT TO 200-DFID-REPORT,
                         200-P-DFID-REPORT.
     WRITE 200-SPOOK-COMMAND-RECORD FROM
           200-COMMAND-LINE 2.
     WRITE 200-SPOOK-COMMAND-RECORD FROM
           200-COMMAND-LINE 3.
     MOVE SPACES TO 200-COMMAND-LINE-1.
     WRITE 200-SPOOK-COMMAND-RECORD FROM
           200-COMMAND-LINE 5.
     MOVE 'EXIT' TO 200-COMMAND-LINE-1.
     WRITE 200-SPOOK-COMMAND-RECORD FROM
           200-COMMAND-LINE-1.
     CLOSE 200-SPOOK-COMMAND-FILE.

A080-END.
     CLOSE 150-SPOOL-INFO-FILE.
     CLOSE 200-SPOOK-COMMAND-FILE.
     CLOSE-250-BANNER-PAGE-FILE.
     GOBACK.
```

PROGRAM: 'CALLSPK'

```
$CONTROL SUBPROGRAM
BEGIN
PROCEDURE CALLSPK;
BEGIN
INTRINSIC CREATEPROCESS, ACTIVATE, KILL;
OWN ARRAY B (0:2);
OWN INTEGER ARRAY IA (0:2) :=1(9,8,0);
OWN LOGICAL ARRAY LA (0:1);
OWN BYTE ARRAY BA1 (0:39) := 1('SO.PUB.PRODW',%15);
OWN BYTE ARRAY BA2 (0:39) := 1('SI.PUB.PRODW',%15);
OWN BYTE ARRAY PRINTPROC (0:39):=
   'SPOOK5.PUB.SYS';
INTEGER ERROR,PIN;
LA(0) :=@BA1;
LA(1) :=@BA2;
CREATEPROCESS (ERROR,PIN,PRINTPROC,IA,LA);
ACTIVATE (PIN,2);
KILL (PIN);
END;
END.
```

PROGRAM: 'WAIT'

```
$CONTROL SUBPROGRAM
BEGIN
INTRINSIC PAUSE;
PROCEDURE WAIT;
BEGIN
REAL A:=6.0E+1;
PAUSE (A);
END;
END.
```

APPENDIX B
SOURCE LISTING FOR ALL JOB STREAMS
JOB STREAM : 'STARTUP'

```
! JOB STARTUP
! OUTFENCE 12;LDEV=137
! HEADOFF 137
! PURGE SO
! PURGE SI
! BUILD SO;REC=-80,,F,ASCII;DISC=100
! BUILD SI;REC=-80,,F,ASCII;DISC=10
! FILE BPAGE;REC=-133,F,ASCII;DEV=137,1,1
! RUN PROCESS1
! EOJ
```

To create the run module 'PROCESS1' all the programs must be compiled into the same USL (User Segmented Library) as follows:

```
:COBOLII MAKEBANN, PROCESS
:SPL CALLSPK, PROCESS
:SPL WAIT, PROCESS
```

Next the USL file 'PROCESS' is prepared with PH (Process Handling) capability.

```
:PREP PROCESS, PROCESS1;CAP=PH
:SAVE PROCESS1
```

JOB STREAM : 'SHUTDOWN'

```
! JOB SHUTDOWN
! FILE STOPSPK;DEV=137,1,1
! LISTF any file;*STOPSPK
! OUTFENCE 1;LDEV=137
! TELLOP HEADON 137
! EOJ
```