

SAS ENGINEERED MVS TOOLS
SOURCE CODE CHANGE REPORT
Tony Melfi, Ohio Bureau of Workers Compensation

ABSTRACT

Being a support person in an IBM shop, I find it helpful knowing what source items have changed recently. Problem resolution time will decrease if the on-call support person can rule out problems relating to a recent source code change. The shop that I work at has a paper based change management process that includes e-mail notification of new or changed items, but this system is subject to human error and is not always published prior to the end of day. Keeping these facts in mind, I thought it would help my teammates and I if we had an electronically generated report that listed all new and changed entries in our production libraries.

INTRODUCTION

This presentation makes use of the SAS procedure called SOURCE, which is listed in the 'SAS COMPANION FOR THE MVS ENVIRONMENT' pages 236 - 246.

The processing option that I used (of PROC SOURCE) gave me the ability to read the directory of a PDS that contained source data members and create a fixed length output record that contained fields such as member name, number of records contained in the member, date created, date changed, and the user ID of the individual who initiated the change. A PDS directory is similar in structure and usage to PC DOS's sub-directories (i.e. C:\MSOFFICE\WINWORD\DOCUMENT1).

The ability to read and create records from the directory of a PDS was of interest to me because the contents of a directory record include date created, date modified, and the initials of the last person that modified this source code.

The shop I work in has three libraries that contain production source code. I listed all those files as input files to SAS. I then listed three output files for the PROC SOURCE command to place the directory output records. The following is the MVS JCL that was input into SAS.

```
//PS010 EXEC SAS
//TMMPROC DD DISP=(SHR),
//          DSN=XXX2.PROCLIB
//TMMJOB DD DISP=(SHR),
//          DSN=XXX2.PRODJCL
//TMSRCE DD DISP=(SHR),
//          DSN=XXX2.SRCELIB
//PERMSAS DD DSN=ZZ.V3M.SASS01.MST,
//          UNIT=SYSDA,
//          SPACE=(CYL,(25,5)),
//          DISP=OLD,
//          DCB=(RECFM=FS,
//          DSORG=PS,LRECL=6144,
//          BLKSIZE=6144)
//DIRFILE DD DSN=ZZ.V3M.SASS02.TRN,
//          UNIT=SYSDA,
//          SPACE=(CYL,(2,1)),
//          DISP=OLD,
//          DCB=(RECFM=FB,
//          LRECL=80,BLKSIZE=0)
//DIRFILEJ DD DSN=ZZ.V3M.SASS06.TRN,
//          UNIT=SYSDA,
//          SPACE=(CYL,(2,1)),
//          DISP=OLD,
//          DCB=(RECFM=FB,
//          LRECL=80,BLKSIZE=0)
//DIRFILES DD DSN=ZZ.V3M.SASS07.TRN,
//          UNIT=SYSDA,
//          SPACE=(CYL,(2,1)),
//          DISP=OLD,
//          DCB=(RECFM=FB,
//          LRECL=80,BLKSIZE=0)
```

The first three steps in this SAS program use the PROC SOURCE command. Each one of the three PROC steps has a different input and output file. A sample of one of the PROC steps follows:

```
PROC SOURCE NODATA NOPRINT
           INDD=TMMPROC
           DIRDD=DIRFILE;
RUN;
```

The 'NODATA' option of the PROC SOURCE command will only process the directory of the PDS. If this option were omitted, the PROC SOURCE command would process every record contained within each member of the PDS. The 'NOPRINT' option of the PROC SOURCE command will inhibit SAS from producing an output list of all member names being processed. I included this option because I wanted an output file. The 'INDD=' option lists the input PDS file to be processed, and the 'DIRDD=' option list the output file that is written. This output file is considered an external file to SAS.

The next two data steps are used to 1) read the previously created data file output from the PROC SOURCE command into a SAS data file, then 2) create another SAS data file that is limited in fields, and identified as a directory entry coming from a particular MVS library. These two data steps are repeated a total of three times using different input and output data file names. The first of these two data steps comes directly from the book; 'SAS COMPANION FOR THE MVS ENVIRONMENT' listed on page 243. An example of these two data steps follows:

```
DATA SASDIRP;
  INFILE DIRFILEP;
  INPUT MEMBER $8.
         TTR    PIB3.
         IND    PIB1. @;
  DATALEN = 2*MOD(IND,32);
  IF (DATALEN = 30)
  THEN DO;
    INPUT VER      PIB1.
          MOD      PIB1.
          BLANK    PIB2.
          CREATE   PD4.
          CHANGED  PD4.
          HH       PK1.
          MM       PK1.
          SIZE     PIB2.
          INIT     PIB2.
          MODL     PIB2.
          USERID   $8.;
    JCREATE = DATEJUL(CREATE);
    JCHANGE = DATEJUL(CHANGED);
  END;
RUN;
```

```
DATA SASDIRP1;
  TYPE='PROC';
  SET SASDIRP
    (KEEP=MEMBER
     SIZE
     JCREATE
     JCHANGE
     USERID);
RUN;
```

The next data step merely copies the three previously created SAS data files into one SAS file. Sample code follows:

```
DATA SASALL;
  SET SASDIRP1
      SASDIRJ1
      SASDIRS1;
  JPLACED=TODAY();
RUN;
```

The next data step sorts the file SASALL into member name and date changed sequence. Sample code follows:

```
PROC SORT DATA=SASALL;
  BY MEMBER JCHANGE;
RUN;
```

At this point in the data step flow, we have created a single SAS file (called SASALL) containing all PDS directory entries from all of the source libraries. This file has been sorted by member name, and changed date. Please note that every record contains the same placed date (JPLACED=TODAY()). When we create the report, we want to see the new or changed entries, how can we see them if they all have the same 'placed date' value? The best way to identify a new or changed member is to compare the contents of a new file to the contents of an old file. The SAS file called SASALL represents our new file. It is a *picture* of our source libraries at this moment in time. We must now have a file to compare against SASALL. This file must be a permanent SAS file that can be compared once a day, every day of the week. The next data step will create the file we need to make the comparison. Please note that this data step should only be executed once. After the initial run, remove this data step, because you will have a permanent SAS data set to be used for comparisons.

Also, please note that the JCL DISP parameter for the file reference PERMSAS must equal 'NEW' on the first execution, and changed to 'OLD' for all subsequent executions. Sample code follows:

```
PROC SORT DATA=SASALL
      OUT=PERMSAS.OLDSTUFF;
      BY MEMBER JCHANGE;
RUN;
```

The next data step plays a key role in providing the information we need to print a change report. This data step has three output files, SASFILE5, SASFILE6, and SASFILE7. At the completion of this step, SASFILE5 will contain all new or changed records, SASFILE6 will contain deleted records, and SASFILE7 will contain all records that have not changed. Please note that the first time this data step is run, all records will be written into SASFILE7 because there will be no differences between SASALL and PERMSAS.OLDSTUFF.

The MERGE statement is at the heart of this data step, and it determines the differences in the two files. The variable 'B' is associated with the file SASALL. When SASALL contributes a record to the merge, the variable 'B' will be set to a '1'. The variable 'A' is associated with the file PERMSAS.OLDSTUFF. When PERMSAS.OLDSTUFF contributes a record to the merge, the variable 'A' will be set to a '1'. When SASALL contains a record that is not in PERMSAS.OLDSTUFF, that record will be written to SASFILE5. In this example B=1 and A=0. When SASALL is missing a record that is contained in PERMSAS.OLDSTUFF, that record will be written to SASFILE6. In this example A=1 and B=0. When both files contribute a record (A=1 and B=1), the contents of the variables, of the second file listed in the merge statement (PERMSAS.OLDSTUFF), will be used in the record written to the output file (SASFILE7). The very first time the merge step is run, the files will be the same, but in the future, when the records are the same, the 'PLACED DATE' will be the placed date from the old file (PERMSAS.OLDSTUFF). Every day when this system runs SASALL will contain the most current library member information, including a 'PLACED DATE' of today's date. If today's record in SASALL is the same as yesterday's record in (PERMSAS.OLDSTUFF), we want to carry forward the placed date that was in yesterday's file. This way, only when a change is encountered, will the placed date be changed. Sample code follows:

```
DATA SASFILE5 SASFILE6 SASFILE7;
      MERGE SASALL (IN=B)
      PERMSAS.OLDSTUFF (IN=A);
      BY MEMBER JCHANGE;
      IF B=1 AND A=0
      THEN OUTPUT SASFILE5;
      ELSE IF A=1 AND B=0
      THEN OUTPUT SASFILE6;
      ELSE OUTPUT SASFILE7;
```

The next data step will copy the SAS file that contains changes (SASFILE5), and the SAS file that contains no changes (SASFILE7), into one file called SASNEW. This file is then sorted into the new permanent SAS file called PERMSAS.OLDSTUFF. Sample code follows:

```
DATA SASNEW;
      SET SASFILE5 SASFILE7;
RUN;

PROC SORT DATA=SASNEW
      OUT=PERMSAS.OLDSTUFF;
      BY MEMBER JCHANGE;
RUN;
```

The last two data steps will 1) Sort the new permanent SAS file into the sequence we want for the printed report, and 2) Print the report only showing changes that have taken place in the last 15 days. The number of days to display on the report can easily be changed by substituting the desired number to subtract from 'TODAY()'. Sample code follows:

```
PROC SORT DATA=PERMSAS.OLDSTUFF
      OUT=SASALL;
      BY JPLACED MEMBER;
RUN;

PROC PRINT N LABEL NOOBS;
      BY JPLACED;
      WHERE JPLACED > (TODAY()-16);

TITLE1 'B W C - S Y S T E M S';
TITLE2 '-----';
TITLE3 'JOBS, PROCS, & SOURCES';
TITLE4 'CHANGED IN THE LAST 15 DAYS';
TITLE5 '-----';

      VAR      MEMBER
              TYPE
              JCREATE
              JCHANGE
              JPLACED
              SIZE
              USERID;
      LABEL JCREATE ='CREATED'
            JCHANGE ='CHANGED'
            JPLACED ='PLACED';

      FORMAT JCREATE JCHANGE JPLACED
      YYMMDD10.;
RUN;
```

 THE FOLLOWING IS A SAMPLE REPORT THAT IS PRODUCED FROM THE PREVIOUS
 SAS PROGRAM

B W C - S Y S T E M S

 JOBS, PROCS, AND SOURCES
 CHANGED IN THE LAST 15 DAYS

----- PLACED=1999-06-07 -----

MEMBER	TYPE	CREATED	CHANGED	PLACED	SIZE	USERID
#WWC181F	JOB	1993-06-17	1999-05-22	1999-06-07	30	MTLH1
#WWC181G	JOB	1993-06-17	1999-06-01	1999-06-07	30	MTLH1
#WWC181H	JOB	1993-06-17	1999-06-05	1999-06-07	30	MTLH1

N = 3

----- PLACED=1999-06-08 -----

MEMBER	TYPE	CREATED	CHANGED	PLACED	SIZE	USERID
DB0FLSCP	JOB	1995-09-18	1999-05-14	1999-06-08	63	XKAK1
DB0FNCCP	JOB	1995-09-18	1999-05-14	1999-06-08	64	XKAK1
DB0FSORP	JOB	1997-05-22	1999-05-14	1999-06-08	63	XKAK1
DWH0005B	SRCE	1998-12-15	1999-06-01	1999-06-08	1179	XSKM1
DWH0030B	SRCE	1998-10-30	1999-06-03	1999-06-08	3307	XSKM1
FRM8030S	SRCE	1997-01-16	1999-05-27	1999-06-08	2276	XMxA1
PCRESTR1	JOB	1995-09-20	1999-06-04	1999-06-08	17	MALL1
PCR5032#	PROC	1999-02-23	1999-06-05	1999-06-08	117	MALL1
PCR5035B	SRCE	1998-12-09	1999-06-01	1999-06-08	650	MALL1
PCVP5051	JOB	1998-02-08	1999-05-03	1999-06-08	151	MVLH1
V3M2654B	SRCE	1999-06-08	1999-06-08	1999-06-08	2644	XCBC1

N = 11

----- PLACED=1999-06-11 -----

MEMBER	TYPE	CREATED	CHANGED	PLACED	SIZE	USERID
DB0PTATP	JOB	1999-05-14	1999-06-09	1999-06-11	104	MTXK1
DB0PTBPP	JOB	1999-06-07	1999-06-09	1999-06-11	104	MTXK1
DB0PTDTP	JOB	1999-06-07	1999-06-09	1999-06-11	104	MTXK1
DB0PTPTP	JOB	1999-06-07	1999-06-09	1999-06-11	104	MTXK1
DB0PTQPP	JOB	1999-05-14	1999-06-09	1999-06-11	104	MTXK1
DB0RCLCP	JOB	1999-06-02	1999-06-03	1999-06-11	63	MDER1
V3M4703B	SRCE	1997-01-08	1999-04-30	1999-06-11	1799	MSFM1
V3M4704B	SRCE	1997-01-03	1999-05-30	1999-06-11	1711	MSFM1
V3M4706B	SRCE	1997-09-08	1999-05-26	1999-06-11	5237	MSFM1
V3M4841B	SRCE	1999-06-03	1999-06-09	1999-06-11	1321	XLMQ1

N = 10

CONCLUSION

The report that is printed on the previous page is a sample of the output of this SAS program. If the BY statement, and the WHERE statement were removed from the PROC PRINT, the report would show all members of all libraries, and the date that those members were placed into their respective PDS production libraries

When viewing a PDS member list in MVS, the date a member was placed is not available, only the created and changed dates are displayed. The report that is generated from this SAS program, adds value to the way MVS displays a PDS library.

A copy of this SAS program, in its entirety, is available on request.

REFERENCES

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries.

Other brand and product names are registered trademarks or trademarks of their respective companies.

“SAS Companion for the MVS Environment”
Version 6, Second Edition, Copyright 1996.

“SAS Procedures Guide” Version 6, Third
Edition, Copyright 1990.

“SAS Language Reference” Version 6, First
Edition, Copyright 1990.

CONTACT INFORMATION

Your comments and questions are valued and encouraged.

Contact the author at:

Tony Melfi
Ohio Bureau of Workers Compensation
30 West Spring Street, Level 18
Columbus, Ohio 43215-2256

Voice – (614) 644-0548
Fax ---- (614) 752-2927
E-mail – Tony.Melfi@BWC.STATE.OH.US