BUILDING AND SUBMITTING JOBS USING TSO AND THE SAS SYSTEM

Jody J. Fromer, The Lubrizol Corporation

Producing reports on demand in a traditional data processing environment usually involves a batch job stream initiated by a user request. The program is written to generate the report and JCL to run the report. The request is then submitted by the user to run the report. Operators submit the job stream and produce the hardcopy output. The report is then distributed to the requestor.

Building systems in the MVS environment using TSO and The SAS System allows us to produce reports literally on demand. Multiple runs for many users can be generated from one generic or skeleton JCL member. The reports are submitted using SAS/AF interactive screens, Macros, and TSO commands. If user input is not required to run the report, then all that is required is Macro, TSO commands, and the JCL member. Let's look at a simple example.

```
//XXXXXXXX JOB !EJJF,'SAMPLE SAS JCL',CLASS=W,
//  MSGCLASS=X,MSGLEVEL=(1,1),NOTIFY=YYYYYYY
//*************************************************
//*                                               *
//*  SAMPLE SUGI RUN                              *
//*                                               *
//*************************************************
//STEP010   EXEC SAS606,WORK='10,10',TIME.SAS606=60,
//*************************************************
// OPTIONS='MACRO MAUTOSOURCE NODATE NONUMBER'
//SASAUTOS  DD DSN=!SU2.SAS606.AUTOLIB,DISP=SHR
//          DD DSN=TEST.SAS.PROGRAMS.DISP=SHR
//*
//SASLOG    DD SYSOUT=M,
//          DCB=(BLKSIZE=1330,LRECL=133,RECFM=FBA)
//SASLIST   DD DSN=&&SAS12,UNIT=SYSDA,DISP=(MOD,PASS),
//          SPACE=(CYL,(10,10)),
//          DCB=(BLKSIZE=264,LRECL=260,RECFM=VBA)
//JJFDATA   DD DSN=RCH1.JJF.SAS.DATA,DISP=SHR
//USERDATA DD DSN=RCH2.!SU0508.JJF.DATA,DISP=OLD
//SYSIN     DD DSN=TEST.SAS.PROGRAMS(JJF020P),DISP=SHR
//*
//
```

Figure 1. Generic/Skeleton JCL (JCL1)

The above figure is a member residing in an ISPF library (PDS). For ease, we'll refer to the member name JCL1. We want to substitute in the userid for the job name, the NOTIFY= statement, and as part of the file referenced as USERDATA. So now, let's examine the macro that makes this all happen.

```
%MACRO PROGJCL1;

X FREE ATTRLIST(ATT1);
X ALLOC FI(PRJOB) DA('TEST.SAS.PROGRAMS(JCL1)') SHR;
X ATTRIB ATT1 DSORG(PS) LRECL(80) RECFM(F B) BLKSIZE(3120);
X ALLOC FI(TEMPJOB) DA(TEMPJOB.DATA) SPACE(1 1) TRACKS CATALOG
        USING(ATT1);

    DATA JOB(KEEP=A);
      XJOB1 = 'SYSJOBID';
      USERID = SYMGET(XJOB1);
    LENGTH A $80;
    INFILE PRJOB;
    INPUT @1 A $CHAR80.;
    IF SUBSTR(A,3,7) = 'XXXXXXX' THEN
       SUBSTR(A,3,7) = USERID;

    IF SUBSTR(A,24,7) = 'DDDDDDD' THEN
       SUBSTR(A,24,7) = USERID;

    IF SUBSTR(A,38,7) = 'YYYYYYY' THEN
       SUBSTR(A,38,7) = USERID;

    DATA _NULL_;
    FILE TEMPJOB;
    SET JOB;
    PUT @1  A $CHAR80.;
    RUN;

       X FREE FI(PRJOB USERDATA);
       X SUBMIT TEMPJOB.DATA;
       X DELETE TEMPJOB.DATA;

%MEND PROGJCL1;
```

Figure 2. JCL build and submit Macro (PROGJCL1)

First, we allocate the skeleton JCL member shown in Figure 1. Then we use the ATTRIBUTE and ALLOCATE commands to define and allocate a file referenced as TEMPJOB. Next is the data step. The SAS dataset we'll create is named JOB. It will have one character variable named X for a length of 80. The infile is the JCL member previously allocated. We input each line of the JCL looking for the "dummy" strings 'XXXXXXXX', 'DDDDDDD', and 'YYYYYYY'. Each of these strings will be replaced by the userid. We get the USERID from the system macro variable SYSJOBID. Once replaced, we use the JOB dataset to load the TEMPJOB file previously defined with TSO commands. The TEMPJOB file is shown in figure 3.

```
//!SU0508X JOB !EJJF,'SAMPLE SAS JCL',CLASS=W,
//  MSGCLASS=X,MSGLEVEL=(1,1),NOTIFY=!SU0508
//*************************************************
//*                                               *
//*  SAMPLE SUGI RUN                              *
//*                                               *
//*************************************************
//STEP010 -  EXEC SAS606,WORK='10,10',TIME.SAS606=60,
//*************************************************
// OPTIONS='MACRO MAUTOSOURCE NODATE NONUMBER'
//SASAUTOS  DD DSN=!SU2.SAS606.AUTOLIB,DISP=SHR
//          DD DSN=TEST.SAS.PROGRAMS.DISP=SHR
//*
//SASLOG    DD SYSOUT=M,
//          DCB=(BLKSIZE=1330,LRECL=133,RECFM=FBA)
//SASLIST   DD DSN=&&SAS12,UNIT=SYSDA,DISP=(MOD,PASS),
//          SPACE=(CYL,(10,10)),
//          DCB=(BLKSIZE=264,LRECL=260,RECFM=VBA)
//JJFDATA   DD DSN=RCH1.JJF.SAS.DATA,DISP=SHR
//USERDATA DD DSN=RCH2.DDDDDDD.JJF.DATA,DISP=OLD
//SYSIN     DD DSN=TEST.SAS.PROGRAMS(JJF020P),DISP=SHR
//*
//
```

Figure 3. TEMPJOB file

Finally, we free the JCL member, submit and then delete the TEMPJOB file. Now, wasn't that simple! But that example is just the tip of the iceberg. Using SAS/AF, we can collect all kinds of user parameters to use in batch jobs. We can route the output to user selected printers, obtain unique distribution ID's, and modify the SYSIN to run any SAS program in the JCL. We can even comment and "un-comment" any JCL lines for even more flexibility. So now, let's examine such an application using SAS/AF. First we display a screen asking for the report selection, the sort variable, the distribution ID, and the printer ID.

```
                        REPORT SELECTION SCREEN
Command ===>

        Place an 'X' next to the desired option and press ENTER.

          _   Print Inventory report sorted by descriptions.

          _   Print Inventory report sorted by codes.

                    Printer Destination is: CR88
                    Distribution: _____________


                                        Hold Queue Commands to Send;
        Valid Printer IDs:                  Print To Output Que:
                                              NP        C
                                              --        -
        CR04 CR88 PF03                        o         a
        PRT5 (RES LASER)                      o         2
        PRT3 (SYSTEM LASER)                   o         r

                            PF3 TO EXIT
```

Figure 4. SAS/AF screen

We capture all the user selections from the SAS/AF screen in global macro variables. Using the report selection and the sort variable as parameters, a macro is executed to write the report to sequential file. Then we run the following macro to build and submit the JCL.

```
%MACRO PROGJCL2(DSN, SORTVAL);
/*************************************************
/*                                               *
/*  SAMPLE SUGI PROGRAM                          *
/*                                               *
/*************************************************
X FREE ATTRLIST(ATT1);
X ALLOC FI(PRJOB) DA('TEST.SAS.PROGRAMS(JCL2)') SHR;
X ATTRIB ATT1 DSORG(PS) LRECL(80) RECFM(F B) BLKSIZE(3120);
X ALLOC FI(TEMPJOB) DA(TEMPJOB.DATA) SPACE(1 1) TRACKS CATALOG
        USING(ATT1);

    DATA JOB(KEEP=X);
      XJOB1 = 'SYSJOBID';
      USERID = SYMGET(XJOBT);
    LENGTH X $80;
    LENGTH DATASET $8;
    INFILE PRJOB;
    INPUT @1 X $CHAR80.;
    IF SUBSTR(X,19,7) = 'XXXXXXX' THEN
                        SUBSTR(X,19,7) = USERID;
    IF SUBSTR(X,3,7) = 'XXXXXXX' THEN
                        SUBSTR(X,3,7) = USERID;

    DHOLD='DIST';
    DHOLD2=SYMGET(DHOLD);
    DHOLD2=LEFT(DHOLD2);
    DHOLD2=TRIM(DHOLD2);

    IF SUBSTR(X,23,15) = 'DDDDDDDDDDDDDDD' THEN
                        SUBSTR(X,23,15) = DHOLD2;

    PRTHOLD='PRT';
    PRTHOLD2=SYMGET(PRTHOLD);
    PRTHOLD2=LEFT(PRTHOLD2);
    PRTHOLD2=TRIM(PRTHOLD2);

    IF PRTHOLD2='PRT3' OR
       PRTHOLD2='PRT5' THEN
       DO;
          IF SUBSTR(X,1,2) = '/*' THEN
          SUBSTR(X,1,3) = '//*';
       END;
    ELSE
          IF SUBSTR(X,15,4) = 'PPPP' THEN
          SUBSTR(X,15,4) = PRTHOLD2;

    HOLD='DSN';
    HOLD2=SYMGET(HOLD);
    HOLD2=LEFT(HOLD2);
    HOLD2=TRIM(HOLD2);
    DSET = 'SORTVAL';
    DATASET = SYMGET(DSET);
    DATASET = LEFT(DATASET);
    DATASET = TRIM(DATASET);

    IF SUBSTR(X,27,8) = 'DDDDDDDD' THEN DO;
          SUBSTR(X,27,4) = HOLD2;
          SUBSTR(X,31,4) = DATASET;
    END;

    DATA _NULL_;
    FILE TEMPJOB;
    SET JOB;
    PUT @1 X $CHAR80.;
    RUN;

TSO FREE FI(PRJOB);
TSO SUBMIT TEMPJOB.DATA;
TSO DELETE TEMPJOB.DATA;
%MEND PROGJCL2;
```

Figure 5. JCL build and submit Macro (PROGJCL2)

The sequential file has the report and sort variable as part of the name. This allows us to use the same JCL to print any report written to a sequential file. We substitute in the userid for the job name, the NOTIFY = parameter, and as part of the sequential file name. There is also logic to assign the printer ID. Once substituted, we use the JOB dataset to load the TEMPJOB file just like the previous example. The TEMPJOB file is shown in figure 6.

```
//ISU0508A JOB IEJJF,'JJF - LUBRIZOL ',CLASS=W,
//   MSGCLASS=X,MSGLEVEL=(0,0)
/*ROUTE PRINT CR88
//STEP01   EXEC PGM=IEBPTPCH
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD DSN=ISU0508.OIDTCODE.DATA,DISP=(OLD,DELETE)
//SYSUT2   DD SYSOUT=*,DCB=LRECL=132
//SYSIN    DD *
         PRINT TYPORG=PS,PREFORM=A
```

Figure 6. TEMPJOB file

Although we allocated a different skeleton JCL member (JCL1), we could have used one generic JCL member and added, deleted, or applied comments to use it in both applications described in this paper.

In conclusion, the SAS system is a very powerful tool which can be used to provide users with the capability to dynamically modify JCL to fit any need or application easily. Using this technique allows us to free up terminal sessions while The SAS System runs in background mode. We minimized required computer resources and design interactive systems which only serve as parameter collection sessions for batch jobs.