Menu-Driven Interfaces Using SAS/FSP* Software
and the SAS* Macro Language

Michael F. Perry, New York State Department of Transportation

SUMMARY

Problem Discussed: Interfacing main frame SAS applications at sites without SAS/AF* or SAS/ASSIST* applications development software.

Intended Audience: Beginning to intermediate SAS programmers who must develop applications without SAS/AF or SAS/ASSIST software.

Suggested Solution: Interfacing SAS applications using SAS/FSP data entry software in combination with the SAS macro language.

Benefits: Allows users to customize program/menu variables in a format that conceals application code, displays detailed user instructions and provides basic input error trapping capabilities.

Outline:
I. INTRODUCTION
II. TWO SIMPLE EXAMPLES
III. CUSTOMIZING SAS/FSP ENTRY SCREENS
IV. ADDING OPTIONAL TESTS AND FEATURES
V. CREATING MENU DRIVEN SYSTEMS
VI. CONCLUSION

I. INTRODUCTION

SAS programmers at sites without SAS/AF or SAS/ASSIST software must either learn third-party application development packages such as ISPF** or physically change "hard coded" statements with new variables each time an application is run. The New York State Department of Transportation (NYSDOT) has licensed SAS, SAS/FSP and SAS/GRAPH* software since 1986 but has been unable to license SAS/AF software due to budgetary constraints. The interface described presents an alternative to learning ISPF or "hard coding" applications by:

1. Establishing a temporary (work) data set of initial program variables or menu choices;

2. Displaying the data set of initial program variables for user modification with PROC FSEDIT and a customized SAS/FSP data entry screen;

3. Converting the user modified initial program variables to SAS macro variables using simple data step "CALL" statements and "SYMPUT" functions, and;

4. Customizing the application with the converted macro variables as necessary.

NYSDOT has successfully used this technique to interface single screen applications as well as small menu driven systems. While no substitute for SAS/AF software, a SAS/FSP interface employs portions of the same SAS Screen Control Language used to drive SAS/AF and SAS/ASSIST software and is far superior to "hard coding" an application. The application programmer may display detailed on-screen instructions, create menus, and establish basic input error trapping capabilities without learning ISPF or exposing program code or macro invocations for user modification.

II. TWO SIMPLE EXAMPLES

Example 1: The first example customizes an application used to browse an existing data set. The code begins by establishing a temporary data set of initial program values to ensure that PROC FSEDIT presents the user with an existing record that is ready to edit. The "SCREEN=" option of the "PROC FSEDIT" statement references a SAS/FSP data entry screen that the application programmer may customize with detailed instructions and basic input error trapping routines. Figures 1 and 2 are fully customized SAS/FSP data entry screens for Examples 1 and 2.*** The macro variable to be created appears first in the SYMPUT function and may duplicate the name of the input data set variable:

Ex. 1 Code: Browsing a Specified Data Set

/* Create an initial variable value. */
DATA TEMP;
  DATASET=" ";

1229
/* Display the initial value on-screen */
/* with SAS/FSP for user modification. */

PROC FSEDIT DATA=TEMP SCREEN=SCRN.BROWSE;
Figure 1: Entry Screen for Example One
---------------------------------------------
Command ==> _

Data Set Browse Selection Screen
---------------------------------------------

Directions: Tab to the entry area below, type the name of the data set to browse from the list below and then press PF3:

Data Set: _______

Data set names/content:
CONTRACT - Contract Level Data.
PROJECT - Fiscal Share Data.
ESTAMT - Engineer Estimate Data.

---------------------------------------------

/* Convert input to a macro variable. */

DATA TEMP; SET TEMP;
CALL SYMPUT(,DATASET,'DATASET');
/* Customize code with macro variable. */

PROC FSBROWSE DATA=DB.&DATASET; RUN;

Example 2: The second example customizes an application used to print all NYSdot contracts with specified minimum values for date of award, awarded dollar amount and number of bidders. SAS system option DQUOTE allows referencing a macro variable (a variable preceded by an "&" symbol) from within quotes. Input data set variables MINAMT and MINDATE create a second set of formatted macro variables for use in customizing the TITLE statement. The code formats MINBIDS as a two position character variable to avoid leading blank spaces caused by converting a numeric data step variable with a default length of eight to a macro variable:

Ex. 2 Code: Printing User Specified Data

/* Create a data set of initial values. */

OPTIONS DQUOTE;

DATA TEMP;
   MINDATE= . ; MINBIDS= ' ' ; MINAMT= . ;
/* Display initial values on-screen */
/* with SAS/FSP for user modification. */

PROC FSEDIT DATA=TEMP SCREEN=SCRN.AWDBID;
INFORMAT MINDATE YYMMD;

Figure 2: Entry Screen for Example Two
---------------------------------------------
Command ==> _

Award Report Modification Screen
---------------------------------------------

Directions: Tab to the entry areas below and type a minimum award date, number of bidders and awarded dollar amount for contracts to be included in the report. Press PF3 when done.

Min. Award Date:
   (E.G.: YYMMD)
Min. No. Bidders:
   (E.G.: 3)
Min. Award $ Amt:
   (E.G.: 5000000)
---------------------------------------------

/* Convert input to macro variables. */

DATA TEMP; SET TEMP;
   CALL SYMPUT('MINDATE,MINDATE);
   CALL SYMPUT('FMTDATE,PUT(MINDATE,
                  WORDDATEI2.));
   CALL SYMPUT('MINBIDS,MINBIDS);
   CALL SYMPUT('MINAMT,MINAMT);
   CALL SYMPUT('FMTAMT,PUT(MINAMT,
                  DOLLARI2.0));
/* Customize code with macro variables. */
/* File TEMP has served its purpose */
/* and may be overwritten. */

DATA TEMP; SET DB.CONTRACT(READ=XXX);
   IF DATE_AWD >= &MINDATE AND
      CONTAAMT >= &MINAMT AND
      NUMBIDS >= &MINBIDS;
PROC PRINT DATA=TEMP;
   VAR CONTRACT DATE_AWD CONTAAMT NUMBIDS;
   TITLE "CONTRACTS AWARDED ON OR AFTER
      &FMTDATE WITH A MINIMUM DOLLAR VALUE OF
      &FMTAMT AND &MINBIDS OR MORE BIDDERS";
   RUN;

The next section discusses modification of SAS/FSP data entry screens with detailed user instructions and basic on-screen input error trapping capabilities.

III. CUSTOMIZING SAS/FSP ENTRY SCREENS

Customizing starts at a default screen the programmer may modify with titles, instructions, examples and basic input error trapping capabilities. For instance, the second example described above would produce the default screen shown in Figure 3 the first time "PROC FSEDIT" displayed the initial program variables for user modification:
To begin customizing, type MODIFY on the command line of the default screen and press the enter key. The "FSEDIT Screen Modification Menu" shown in Figure 4 will appear after a few seconds:

**Figure 4: SAS'FSP Modification Menu**

---

Select Option

1. Information about screen modification
2. Screen Modification
3. Field Identification
4. Field Attributes
5. Parameter Modification
---

Option two allows the programmer to customize the basic screen using line editing commands similar to those used during SAS Display Manager sessions. The programmer may completely rearrange the location of input variables, type detailed instructions or even replace the basic variable name prompts with longer and more descriptive prompts. Option three, or pressing PF3 after completing screen modification, allows identification of the area on the modified screen where a user will input a variable. Options four and five manage screen colors and basic error trapping capabilities to verify minimum, maximum, required, capitalized or initial input values. Pressing PF1 after selecting any option provides excellent on-screen help facilities. To revise a previously modified screen, retype MODIFY on the command line of the screen to be revised and press enter.

Since most users assume SAS/FSP screens are normal menus driven by the enter and arrow keys, SAS/FSP interface screens should include instructions reminding the user to tab to the entry areas and press PF3 to submit their choices. There is also a slight chance the user could become confused by accidentally issuing a SAS/FSP command to add or repeat a record. So far, NYSDOT has experienced a few instances of users forgetting to tab and press PF3 and no instances of users accidentally creating additional records.

The fully modified SAS/FSP screen file must be stored in a manner accessible to the application code it interfaces.

While the SAS system usually handles such tasks automatically, problems may result if the programmer allows access to the application code without also allowing access to its modified SAS/FSP screen files. An application which loses access to its modified screen will produce a default screen once again. Restoring access to the appropriate screen file corrects the situation the next time the application is run.

**IV. ADDING OPTIONAL TESTS AND FEATURES**

Formatting a SAS/FSP interfaced application as a simple SAS macro allows testing and supplementing user input with the SAS macro language. For instance, browse Example 1 could supplement a data set read password based on user input as shown below in Modification (Mod.) 1:

**Mod. 1: Optional Macro Tests & Features**

```sas
%MACRO begins macro compilation. */
%Establish/edit variable as before. */
%MACRO BROWSE;
DATA TEMP;
DATASET='.
PROC FSEDIT DATA=TEMP SCREEN=SCRN.BROWSE;
/* The RUN statement completes the */
/* SYMPUT function prior to execution */
/* of the macro %IF test. */
DATA TEMP; SET TEMP;
CALL SYMPUT('DATASET',DATASET);
RUN;
/* Test and supplement user input. */
%IF &DATASET = ESTAMT OR &DATASET = ITEMPEST */
%THEN %LET PW=YYY;
%ELSE %LET PW=XXX;
/* Customize code with macro variables */
/* and close and call the macro. */
PROC FSBROWSE DATA=DB.&DATASET(READ=&PW);
RUN; %MEND BROWSE; %BROWSE
```

**V. CREATING MENU DRIVEN SYSTEMS**

A macro format also allows interfacing applications with a common menu. The following code establishes the menu shown in Figure 5 to determine which example the user wishes to execute:

**Mod. 2: Menu for Examples 1 and 2**

```sas
/* Execute the %DO-%UNTIL loop until a */
/* blank entry is detected from a */
/* previous loop execution. */
```
%MACRO MENU;
%DO %UNTIL(&OPTION= );
  /* Create a data with an initial menu */
  /* choice. */
DATA TEMP; OPTIONS DQUOTE;
   OPTION= ' '; /* Display the initial menu choice for */
   OPTION= ' '; /* user modification with SAS/FSP and */
   OPTION= ' '; /* the customized screen shown below. */
PROC FSEDIT DATA=TEMP SCREEN=SCRN.MENU;

Figure 5: Example SAS/FSP Menu
------------------------------------------
Command --> _
Example Application Menu
------------------------------------------

Directions: Tab to the entry area below,
you must type the one letter code
the task you wish to perform and then press PF3:

Task to Perform: _

Valid Letter Codes: B = Screen Browse.
A = Award Report.
   (blank) Exit.

Convert input to macro variables. */
DATA TEMP; SET TEMP;
   CALL SYMPUT('OPTION',OPTION);
RUN;

/* If user input is "B," execute the */
/* following browse example code. */
/* Otherwise, skip to the next %END. */
/* (Note: For full commenting of the */
/* example code, refer to the previous */
/* discussions of the examples. */
%IF &OPTION = B %THEN %DO;
DATA TEMP;
   DATASEt=' '; PROC FSEDIT DATA=TEMP SCREEN=SCRN.BROWSE;

DATA TEMP; SET TEMP;
   CALL SYMPUT('DATASET',DATASET);
RUN;

/* If &DATASET = ESTAMT OR */
/* &DATASET = ITEMPEST */
%THEN %LET PW=YYY;
%ELSE %LET PW=XXX;
PROC FSBROWSE DATA=DB.&DATASET(READ=&PW); RUN;
%END; /* Close the option B %DO group. */

/* If user input is "A," execute the */
/* the following browse example code. */
/* Otherwise, skip to the next %END */
%ELSE %IF &OPTION = A %THEN %DO;
DATA TEMP;
   OPTIONS DQUOTE;
   DATASEt=' '; MINBIDs=' '; MINAMT= ' ;
PROC FSEDIT DATA=TEMP SCREEN=SCRN.AWDBID;
   INFORMAT MINDATE YYYYMMDD.;

DATA TEMP; SET DB.CONTRACT(READ=XXX);
   IF DATE_AWD >= &MINDATE AND 
   CONTAAMT >= &MINAMT AND
   NUMBIDS >= &MINBIDS;
PROC PRINT N DATA=TEMP;
   TITLE "CONTRACTS AWARDED ON OR AFTER "
   &FMTOATE WITH A MINIMUM DOLLAR VALUE OF
   &FMTAMT AND &MINBIDS OR MORE BIDDERS";
RUN;
%END; /* Close the option A %DO group. */
%END; /* Close loop the %DO-%UNTIL group. */
%MEND MENU; /* Close macro MENU. */
%MEND MENU; /* Call macro MENU. */

More complex menu driven systems are possible and best designed using an
autocall SAS macro library and SAS system
option MAUTOSOURCE. For instance, Mod. 3
shows how the MAUTOSOURCE option
could streamline the above menu code by
storing the example code as individual
members (subroutines) entitled BROWSE and
AWDBID in an autocall (MACLIB) library and
referencing them as %BROWSE and %AWDBID
when and if necessary:

Mod. 3: Using an Autocall Macro Library

%MACRO MENU;
%DO %UNTIL(&OPTION= );
DATA TEMP; OPTIONS DQUOTE;
   OPTION= ' '; PROC FSEDIT DATA=TEMP SCREEN=SCRN.MENU;

DATA TEMP; SET TEMP;
   CALL SYMPUT('OPTION',OPTION); RUN;
/* Test user input and call the macro */
/* chosen from the autocall library. */

1232
VI. CONCLUSION

Programmers at sites without SAS/AF or SAS/ASSIST applications development software may create single and multiple screen user interfaces using SAS/FSP data entry software in combination with the SAS macro language. Specifically, the technique establishes a temporary SAS data set of initial program variables, displays the initial values for user modification with SAS/FSP data entry software and retrieves user choices from the modified data set as macro variables used to customize the application.

The technique works best with single screen applications or small menu driven systems progressing no more than two to three screens beyond an opening menu. Specific limitations include:

1. A few seconds delay each time PROC FSEDIT displays an input screen;
2. A slight chance users may become confused by accidentally issuing a SAS/FSP software command to create an additional entry record, and;
3. A data entry screen format which requires users to Tab to the designated entry area and press PF3 when done.

However, for sites without SAS/AF or SAS/ASSIST applications development software, the technique is preferable to exposing actual program code for user modification and accesses many otherwise unavailable capabilities of the SAS Screen Control Language. NYSDOT has successfully expanded the basic concepts of this presentation to interface relatively complex single and multiple screen applications.

For Additional Information Contact:

Michael F. Perry
NYS Department of Transportation
Computer Services Bureau
1220 Washington Ave.
Albany, NY 12223
Telephone (518) 457-5575

* SAS, SAS/FSP, SAS/AF, SAS/ASSIST and SASGRAPH are registered trademarks of SAS Institute Inc., Cary, NC, USA.
** ISPF is a trademark of International Business Machines Corporation.
*** Please note that this presentation purposely limits all SAS/FSP screen examples to forty columns to match the forty column format of the SUGI-15 Proceedings. Actual SAS/FSP data entry screens are eighty columns wide and much neater than the manually retyped examples of this presentation.

Acknowledgements: Thanks to Carol Fryc and Ted Mulberry for proofreading, Bob Holland and Ken Harper for paper layout.