Paper 193-26

Farewell to Documentation Despair!
The Development of a Documentation System Using FSEDIT and FSBROWSE

Amy M. Carey, Howard M. Proskin & Associates, Inc., Rochester, NY

ABSTRACT

Careful documentation is a universal need of critical importance. This paper presents a documentation application, developed in SAS® version 8.0 running on a Microsoft Windows NT platform, which can be constructed in a few hours by a beginning to intermediate level SAS user. The FSEDIT procedure provides a means of entering new information into the system, using a customized entry screen, which is stored on a shared data network. A few lines of SAS Component Language (SCL) were used in the customization of the screen to automatically enter values for a number of variables (*name*, *date*, etc.). The FSBROWSE procedure was employed for viewing information. As the data set containing the documentation information becomes increasingly large, it will be necessary to limit the viewing to specific entries. Thus an FSEDIT window was used to specify which entries should be displayed. For quick access to the system, the tool bar was customized to include a button which submits the editing program, and one which submits the viewing program. Since it is capable of considerable customization and expansion, the basic documentation system developed here can be easily adapted to various environments.

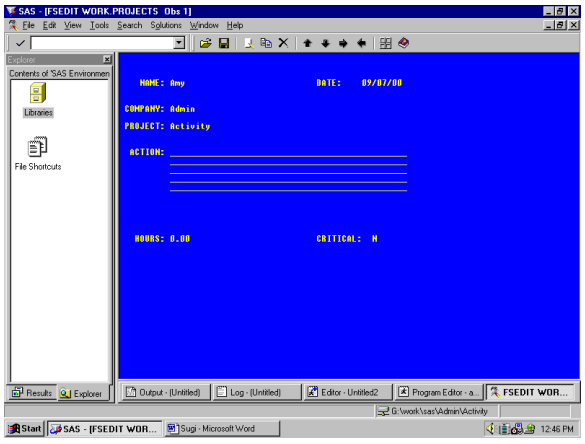
INTRODUCTION

As the size and scope of projects grow, so does the number of people involved. Therefore, it becomes increasingly crucial to have thorough documentation. As a small consulting firm, we were able to utilize the SAS system to accomplish this task. All of our users are working on PCs running Windows NT and are networked to a central data server. This application was developed to quickly document progress on projects that could easily be viewed by anyone on the network. One convenience of this system is that multiple users are capable of viewing this documentation simultaneously. Also, since most of our tasks are performed within SAS, it was fitting to have the ability to document within SAS as well.

SYSTEM FEATURES

ENTERING DOCUMENTATION IN SAS

The FSEDIT window is utilized to enter the documentation. All documentation entered through this means is stored in a SAS data set. The documentation entry screen is shown below.



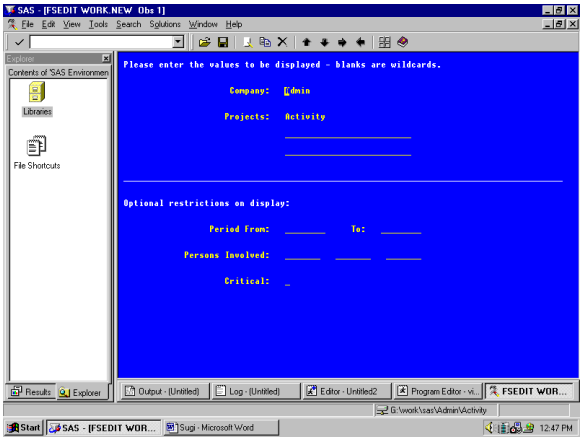
The variables *name*, *date*, *company*, and *project* are automatically entered when the program is submitted (see "Customizing the Screens", below). This was implemented to reduce the fields that the user would have to enter manually. The *name* and *date* fields are protected and therefore cannot be overwritten by the user. The *company* and *project* fields are extracted from the current directory (in this window the directory is g:\work\sas\admin\activity). These fields can be overwritten in case the user prefers to document all of their activities at the end of the day, or if the directory was not named according to this convention. All fields are required, yet since there are default values for every field except *action*, this is the only field that the user must fill in each time.

ENTERING DOCUMENTATION IN MICROSOFT EXCEL

Since not all of our tasks are performed in SAS, it was preferable to integrate an application that would enable entry outside of the SAS system. Therefore an Excel spreadsheet was created with variables corresponding to those in the SAS data set. The information is recorded in this file, which is stored on the data server and thus is accessible to all users that are connected to the network. This file is then imported into SAS and merged with the master SAS documentation data set upon execution of the viewing program (see "Appendix 2", below).

VIEWING INFORMATION

The FSBROWSE window is employed for documentation. The user is able to specify values of the parameters to be viewed such as specific persons, days, projects and companies. By default, the FSBROWSE application will display all entries for the current project (determined by the directory name). The screen to specify viewing parameters follows.



DEVELOPMENT OF THE SYSTEM

GETTING STARTED

Initially, a data set "blank" was created which specified the variable names, lengths, formats and informats that would be used for entering project information. An identical data set, "master" was created to store all of the project information. A similar data set "blnkpref" was created containing information to be used for entering viewing preferences.

All SAS programs, data sets and display catalogs were stored on the network so that they would be accessible to all users. Each user included a line in their autoexec.sas file to assign the libname "activ" to this directory on the network.

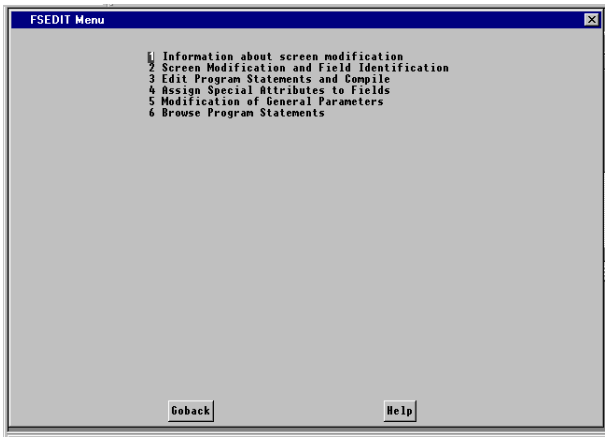
CUSTOMIZING THE SCREENS

The majority of the development was done within the FSEDIT Menu. The following code was used to create the entry screen for adding documentation.

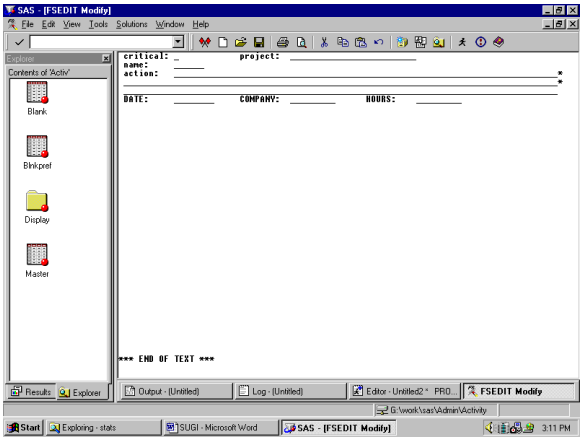
```
proc fsedit data=activ.blank modify
  screen=activ.display.projscr.screen;
run;
```

The empty data set “blank” is used to create this screen. The customized screen “projscr” is stored in the “activ” library in the “display” catalog. For this documentation system, three screens were created; “projscr” for entering documentation, “prefscr” for entering viewing preferences, and “viewscr” for viewing documentation.

When the FSEDIT procedure is implemented with the modify option, or when the modify command is issued within an FSEDIT session, the following screen is displayed.



The second menu item enables you to design the screen. As is seen below, the variables will be listed on the screen with data entry lines following.



The text colors can be changed by pressing the ESC key, followed by the which letter represents the desired color (W for white, Y for yellow, etc.).

After designing the screen, close the FSEDIT window or type “end” on the command line. At this point, the variables will need to be identified so that the SAS system knows the location of each variable on the newly designed screen. This is accomplished by pressing enter on the data entry line for each variable, one at a time, following the prompts at the bottom of the screen.

Further modifications can be made to the screen under menu items 4 (to set preferences for each variable such as protection, default values, required values, etc.) and 5 (to modify the screen colors, enable/disable the deletion of observations, establish a password to protect the screen from modifications, etc.).

The following SCL program statements were added in the entry application (menu item 3) so that some values would be automatically entered.

```
init:
tmp=filename('xfold','.');
sid=dopen('xfold');
dirname=dinfo(sid,'DIRECTORY');
projname=reverse(scan(reverse(dirname),1,'\'));
project=compress(projname);
compname=reverse(scan(reverse(dirname),2,'\'));
company=compress(compname);
tmp=dclose(sid);
name=symget('who');
date=today();
hours=0.00;
critical='N';
return;
main:
return;
term:
return;
```

The init section of this code is executed prior to the display of each observation. The values of *company* and *project* are extracted from the current directory name, according to our company’s standard naming conventions. The value of *name* is pulled from the macro variable *who*, which is assigned at the beginning of the add documentation program (see “Appendix 1”, below). The system date is assigned to be the initial value for the *date* variable.

Initial values for *hours* and *critical* are also assigned here.

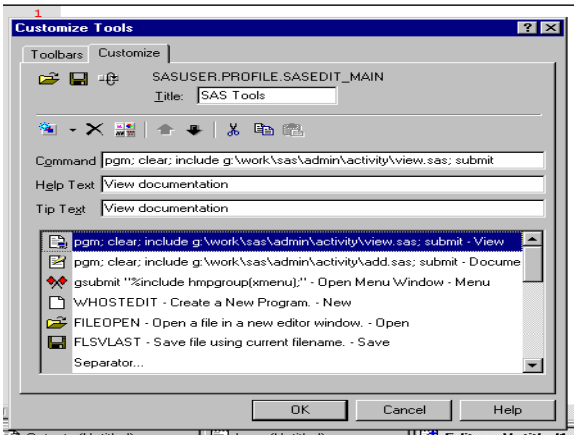
The main and term sections are required, but for this application there is no specific need for them. The main section is executed as each observation is updated and the term section is executed before moving to the next observation.

PUTTING IT ALL TOGETHER

The program in “Appendix 1” is used to add observations to the master documentation data set. This program assumes that the data sets “blank” and “master”, which were established prior to the first execution of this program (see “Getting Started”, above), exist within the “activ” library. A second program, which can be found in “Appendix 2”, is used to enable the user to view the data.

USER-FRIENDLINESS

Rather than having each user manually recall and submit the program each time he/she needs to add documentation or view the documentation for a particular project, icons were created on the toolbar. As seen below, this is accomplished by adding a tool under the “Customize toolbar” tab from the Tools-> Customize menu and adding the appropriate code on the “command” line.



Once a button is created, the documentation is a snap. When the button that corresponds to these commands is clicked, a program editor window will be opened; if one is already opened the window will be activated. The contents of the program editor window will be cleared, the program will be included in the window, and the program will be submitted, thus implementing the corresponding system.

POSSIBLE EXTENSIONS

At this point, the program is quite functional. It provides a quick and easy means of documenting and viewing information for each specific project. However, there are infinite possibilities with this system.

The modules can be easily modified and expanded. Larger corporations might want to adapt the programs to better suit their needs, adding more variables to the add documentation program, or expanding the viewing program to enable more preferences to be specified.

The possibilities for small businesses include incorporating invoice information into the system. When invoices are sent out, the company, project, period of time billing for and the date sent could be recorded. This viewing screen could be adjusted so that one could view information on a project since the last invoice (to determine hours yet to be billed), or even to determine how many billable hours each employee is working. The possibilities are endless.

CONCLUSION

With simple FSEDIT and FSBROWSE applications, the imperative task of documentation can be one of the quickest tasks of the day. Establishing the system presented here is a relatively quick project which provides users with a very user-friendly documentation environment.

APPENDIX 1 – ADDING INFORMATION

```
/* Read in data set with blank variables, pull
username from the system */
data projects;
  set activ.blank;
  %let who=%sysget(username);
run;

/* Add new info into the blank variables */
proc fsedit data=projects
  screen=activ.display.projscr.screen;
run;

/* Append new info to previous data set */
proc datasets;
  append base=activ.master data=projects;
run;

/* Delete data set to prep for next use */
proc datasets;
  delete projects;
run;
```

APPENDIX 2 - VIEWING INFORMATION

```
/* Read in data set with blank variables */
data new;
  set activ.blnkpref;
run;

/* Add new info into the blank variables */
proc fsedit data=new
  screen=activ.display.prefscr.screen;
run;

/* Keep last obs preference data set only (i.e.
the obs just entered by the user) */
data pref;
  set new;
  firstobs=999999;
run;

/* Import data from Excel spreadsheet */
proc import datafile=
  "g:\work\sas\admin\activity\activity.xls"
  out=temp
  dbms=excel97 replace;
run;

/* Fix date before merging with SAS data */
data fixdate;
  format newdate mmddyy8.;
  set temp;
  newdate=datepart(date);
  drop date;
run;

/* Remove extraneous observations */
data prepxls;
  set fixdate;
  date=newdate;
  if company=' ' then delete;
  drop newdate;
run;

/* Merge SAS and Excel data */
data all;
  set activ.master prepxls;
run;

/* Merge current prefs with documentation */
data viewpref;
  if _n_=1 then set pref;
  set all;
run;

/* Flag obs matching user specified prefs,
display all obs if nothing was specified */
data view;
  set viewpref;

  if (from=. and to=.) then tdate=1;
  else if (date ge from and date le to) then
    tdate=1;
  else tdate=0;

  if person1=person2=person3=' ' then tprsn=1;
    else if (name=person1 or name=person2 or
    name=person3) then tprsn=1;
  else tprsn=0;

  if project1=project2=project3=' ' then
    tproj=1;
  else if (project=project1 or project=project2
    or project=project3) then tproj=1;
  else tproj=0;

  if comp1=' ' then tcomp=1;
  else if company=comp1 then tcomp=1;
  else tcomp=0;

  if imprtnt=' ' then tcrit=1;
  else if imprtnt=critical then tcrit=1;
  else tcrit=0;
run;
```

```
/* View information specified */
proc fsbrowse data=view
  screen=activ.display.viewscr.screen;
  where tdate=tprsn=tproj=tcomp=tcrit=1;
run;

/* Delete data set to prep for next use */
proc datasets;
  delete projects;
run;
```

REFERENCES

SAS/FSP Software: Usage and Reference, Version 6, First Edition.

SAS Screen Control Language: Reference, Version 6, First Edition.

ACKNOWLEDGMENTS

Special thanks to Edie Hogan for her patience and help with the development of this project. I would also like to thank William Murphy for his SAS guidance, and Howard Proskin for his helpful suggestions and encouragement.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Amy M. Carey
Howard M. Proskin & Associates, Inc.
2468 East Henrietta Road
Rochester, NY 14623
Work Phone: 716-359-2420
Fax: 716-359-0465
Email: acarey@hmproskin.com