ABSTRACT

The College of Forestry at Stephen F. Austin State University, maintains a data base of climatological information gathered from a weather station that has been maintained on a daily basis since 1976. Using data collected, key pieces of information were hand calculated and then data were entered into a Honeywell CP6® using the line editor. This resulted in a large file, full of errors, requiring an experienced programmer to extract useful data. This paper presents an FSEDIT® application developed to streamline data entry and eliminate errors associated with hand calculations while creating a more useful and attractive interface. This application was developed in SAS/FSP using FSEDIT and Screen Control Language.

INTRODUCTION

The FSEDIT option of FSP offers a quick and easy way to enter, edit and view data. Once learned, a data entry screen can be created and operational in a matter of minutes. Besides its ease of use, FSEDIT is also a very expandable option. Used in conjunction with Screen Control Language (SCL), it can be enhanced to meet virtually any performance need.

BACKGROUND

The College of Forestry at Stephen F. Austin State University, Nacogdoches, Texas has maintained a weather station since 1976. From this station daily
climatological data has been collected for reporting to the National Weather Service, Texas Water Development Board and to various mass media in the area. From its beginning, certain pieces of information were hand calculated and then entered into an outdated Honeywell CP6 using the line editor. During the mid-seventies a FORTRAN program was written to summarize the data by month. However, no one has been available to modify or maintain the code since it was written. Consequently, for years nothing has been done with this valuable resource after its initial reporting use.

We were first enlisted to help summarize the weather data last year and this is when the poor condition of the database was learned. It was decided that a new system of entering and storing the data was needed. SAS, due to its flexibility, was the logical choice. As a result, a data entry application to facilitate ease of editing and viewing was developed. In time, as we gain experience, this project will hopefully grow into a weather information system, where summaries, graphics and data entry are integrated into a more robust application.

PROBLEM

The over-riding goal during its development was to make the application easy for inexperienced users to access. Using FSEDIT it is possible to mimic a data recording sheet. Many times people tend to create a simple yet effective recording sheet for use in the field, but lose the simplicity when transcribing the data from paper to the computer. A common occurrence is when a person has developed their own field tally sheet that is simple and easy to use with flowing and logical entry points, then fall back on the old row and column technique for computer entry. Thus was the case with the weather data. A nice flowing tally sheet (Appendix A.) was being transcribed into a single line with no spaces between entry points (Figure 1).
Reading in the original weather data text file to a SAS data set should have been a simple data step. However, it took several exasperating attempts to get it right. Due to the layout of the original data set there were numerous errors. Over the years, numbers and characters had been mistyped, spaces had been added or left out, and unnecessary decimals entered. What was considered as the main source of error in the data set was the daily evaporation and mean wind speed. These two key pieces of information were being hand calculated, written on the data sheet and then entered on the computer. These extra steps invited a higher possibility for errors. The use of SCL within the data entry screen would alleviate this problem.
**APPROACH**

**Data entry screen**

The creation of a custom data entry application with the look and feel of the paper form was achieved by running the FSEDIT procedure. To initiate the procedure you must not only specify the data set but also assign a screen name as shown in Figure 2. where SASUSER is the libref, WEATHER is the catalog name and WEATHER.SCREEN is the screen name.

```
PROC FSEDIT DATA=SASUSER.WEATHER
SCREEN=SASUSER.WEATHER.WEATHER.SCREEN;
RUN;
```

Figure 2. SAS code to initiate FSEDIT.

This opens an FSEDIT window with a default screen arrangement. An example is shown in Figure 3.

![Figure 3. Example of default FSEDIT screen.](image-url)
In order to arrange the screen to the desired format the MODIFY command was issued. You would then be presented with a series of options, one of which is "Screen modification and field identification." After selecting this option you are returned to the data entry window where you are able to customize the screen to suit your needs. After the screen modification is completed you are then asked to identify the location of each variable field. Figure 4. shows the customized weather data entry screen. The design of this screen is a condensed version of the paper form. Some of the data points were not used, such as the conditions at the time of observation.

Figure 4. Customized weather data entry screen.

Several field attributes are available to assist users in avoiding errors during data entry or editing. A field can be set up to: enter an initial value when an observation is added; minimum and maximum limits can be set for a given field;
fields can be protected from editing or colors assigned to fields as desired.

**SCL program**

The creation of the data entry screen was only half the battle. SCL must be utilized for the application to calculate daily evaporation and mean wind speed. In order for an SCL program to run it must know when to execute its specific components. The implementation of our SCL program is controlled by four executable sections:

1. **FSEINIT**: Opening the SAS data set for SCL to access and also assigns a data set identifying name.
2. **INIT**: Collects the previous day's reading for evaporation and wind and calculates the evaporation and mean wind speed if the previous days values had been modified.
3. **MAIN**: Calculates the evaporation and average wind speed if a field has been modified.
4. **FSETTERM**: At the end of the session this closes and resets the SAS data set to clear any deleted observations.

Until this application was implemented the hand calculated values for mean wind speed and evaporation had been entered into the computer, but not the actual readings that are used in the calculations. Therefore, we needed a check so that those two variables would not be calculated by SAS unless all the information was present. This was accomplished by using an "if-then" statement to test if the date was greater than the implementation date in which the required data had been entered. If this was true then the calculations would be performed, otherwise, it would be skipped.

The difficult part of the development process was to have SAS calculate the two variables. What would seem to be a simple and straightforward problem turned into one that even SAS support was stumped on. In fact, after two weeks
they called with a solution that we had stumbled upon a few days earlier. It was that event which prompted us to submit this paper.

In order to calculate both the mean wind speed and the evaporation it is necessary to subtract the current day's reading from the previous day's reading. This was accomplished by using several SCL functions. If you are going to be calculating a value using a current and previous observation it would seem fairly obvious you would need to know which observation you were working with. This could be done using the CUROBS function which returns the number of the current observation. A problem arises however, when working with a new observation. New observations created with the ADD command are not actually added to the data set until you save, end or move to another observation. Therefore, a new observation would not have an observation number. To overcome this we used the OBSINFO function to test if the currently displayed observation was a new observation. If so, then the "NOBS" option of the ATTRN function would be issued to give us the total number of observations except for the new one. This returned value would then be the previous observation. If the current observation is not a new one, then we would use CUROBS function and subtract one from the result to give us the previous observation number.

Once we had the correct observation numbers we needed to collect the necessary values from the previous observation. This was done with the FETCHOBS function which reads the specified observation from the data set into the data set data vector (DDV). The actual values must then be extracted from the DDV. This was a two step process in which the variable number or the position of the variables in the SAS data set are returned with the VARNUM function. From there the contents of the specified variables could be accessed using the GETVARN function. This function reads the value of a numeric variable from the selected observation in the DDV and writes it to an SCL variable. Once all necessary values
were stored in the SCL data vector the final calculations could be made. In our case, a current day's reading was subtracted from the previous day's reading. SCL code may be found in Appendix B.

CONCLUDING REMARKS

Hopefully, this paper shows how easy it is to create your own data entry applications that can let people break free from the old row and column ball and chain. With the additional use of SCL, many of the errors associated with hand calculations can be avoided or eliminated.

REFERENCES


ADDITIONAL INFORMATION

For additional information, Please contact:

Gordon Holley  
College of Forestry  
Box 6109, SFA Station  
Nacogdoches, TX 75962-6109  
(409)468-1348  
gholley@sfasu.edu

Leslie Dale  
College of Forestry  
Box 6109, SFA Station  
Nacogdoches, TX 75962-6109  
(409)468-1349  
ldale@sfasu.edu
Appendix A

PAST 24-HOUR WEATHER
SFA FORESTRY WEATHER STATION
TDWR EVAPORATION STATION #26
NACOGDOCHES, TEXAS

DATE OF REPORT: MONTH _______ DAY _______ YEAR _______

TEMPERATURE

High of ______________ Degrees at ___________________ AM/PM
Low of ______________ Degrees at ___________________ AM/PM
Temperature at observation was ___________ Degrees at ___________ AM/PM.

RELATIVE HUMIDITY

High of ______________ % at ___________________ AM/PM
Low of ______________ % at ___________________ AM/PM
Humidity at observation was __________ % at __________ AM/PM.

PAST 24-HOUR PRECIPITATION

__________ inches of Rainfall __________; Snow __________; Sleet

PAST 24-HOUR WIND MOVEMENT

Today's reading: ____________________
Less Yesterday's reading - _______________
Total daily wind movement was = _______________
Average hourly wind speed _______________ mph
Wind direction at observation was out of the ____________

PAST 24-HOUR EVAPORATION

Yesterday's reading ______________
Today's reading - ______________
Plus Precipitation + _______________
Total day's evaporation was = ______________

RECORER AND TIME OF REPORT
APPENDIX B

*/
*******************************************************************************/
* SCL code to calculate means wind speed and evaporation.
* 1996
*
* Variables used in SCL
*
* Number Name Type Description
*******************************************************************************/
**********************************************************
1 refill Num Evap pan refill
2 wind Num Current wind reading
3 Evapor Num Current pan reading
4 Rain Num 24 hr precipitation
5 Mean_wv Num 24 hr mean wind speed
6 Evap Num 24 hr evaporation
**********************************************************
/;

FSEINIT:

*/
*******************************************************************************/
* The FSEINIT section opens the "weather" data set in input mode for updating.
* Libref = sasuser
*******************************************************************************/
*/
DSID=OPEN('sasuser.weather','i');
RETURN;

INIT:

*/
*******************************************************************************/
* This section loops through and checks for a new observation, entries in the EVAPOR and WIND fields and calculates EVAP and MEAN_WV
*******************************************************************************/
*/
/* Checking for new or old observation /*;
if obsinfo('NEW') then do;
oldobs=attrn(dsid,'nobs');
obsnum=obsinfo('curobs');
end;
else do;
obsnum=curobs();
oldobs=obsnum - 1;
end;
/* Getting previous days observation */
if oldobs >= 1 then do;
  loop';
  rc=fetchobs(dsid,oldobs,'ABS');
  if rc=0 then do;
    x = varnum(dsid,'wind');
    y = varnum(dsid,'evapor');
    z = varnum(dsid,'refill');

    prewind = getvarn(dsid,x);
    prevapor = getvarn(dsid,y);
    prefill = getvarn(dsid,z);

  /* Performing calculations */
  if mean_wv EQ . then mean_wv=((wind - prewind) / 24);
  precip = rain;
  if precip = . then precip = 0;
  if obsnum > 7258 then
    if prefill = . then
      evap=((prevapor - evapor) + precip);
    else if prefill > 0 then
      evap=((prevapor - evapor) + precip);
    else do;
      _msg_='Previous observation deleted calculations will not be performed';
      mean_wv= .;
      evap= .;
    end;
  end;
RETURN;

MAIN:

/*
 ****************************************************
 * This section loops through and checks for modifications to the EVAPOR, WIND and REFILL fields and calculates EVAP and MEAN_WV
 *****************************************************/

if modified(wind) and wind NE . then
  mean_wv=(wind - x) / 24;
precip = rain;
if precip = . then precip = 0;
if modified(evapor) or modified(rain) and evapor NE . then
  if x3 = . then
    evap=((x2 - evapor) + precip);
  else if x3 > 0 then
    evap=((x3 - evapor) + precip);
  end;
RETURN;
TERM:
   */
   *************************************
   * Not used for this code, only here *
   * because of warning msg in log    *
   * otherwise                       *
   *************************************
   */
RETURN;

FSETERM:
   */
   *************************************
   * This section closes the data set and submits  *
   * code to reset the data set to clear any deleted *
   * observations                             *
   *************************************
   */;

DSID=CLOSE(dsid);
submit;
   data sasuser.weather; set sasuser.weather; run;
endsubmit;

RETURN;