

Using SAS/AF[®] to Create Data Entry Screens with Multiple Comment Fields Allowing for Unlimited Amounts of Text in Each Field

Robert S. Matthews, University of Alabama at Birmingham, Birmingham, AL

Abstract

This paper describes a process, using SAS/AF software, for creating data entry screens that can contain multiple comment fields. An unlimited amount of text information can be stored in each field and there can be an unlimited number of comment fields for each observation. The comment fields are defined as 'Extended Text Entry' widgets in an AF Frame screen. The text information in the comment fields is stored in a SCL List that is then saved to, and retrieved from, a SAS catalog entry. A SAS dataset is used in conjunction with a SAS catalog to provide a complete data entry system. The intended audience for this paper should have an intermediate level of knowledge about the Base SAS System as well as proficiency using SAS/AF and Screen Control Language.

Introduction

The process of creating unlimited text entry fields was crucial to implementing an exposure estimation system we developed for managing a large collection of task exposure information. In addition to entering all of the actual exposure data, the investigators wanted to be able to record extensive comments about many of the different exposure components. Microsoft Word documents already existed which contained some task comments, other comments existed in Excel spreadsheets, and others would need to be entered into the system manually. Another design criteria was to be able to enter any amount of textual information into the comment fields. Placing a subjective limit of 10, 20 or even 100 lines of text was not acceptable because some documents were several hundred lines in length.

We explored the feasibility of using SAS datasets to store the comment fields, but this approach had several significant weaknesses. One problem with using standard SAS character variables was the fact that each character variable would have to be defined in advance. Another disadvantage was the amount of wasted space that would occur with dozens or hundreds of predefined character variables, each of which

would need to be 100 or more bytes in length. The major reason we discounted this approach was the fact that we needed to have the ability to enter an unlimited amount of text in each comment field. This design criteria forced us to come up with a novel solution for implementing unlimited text entry fields.

Solution

The solution we developed uses 'Extended Text Entry' widgets in a SAS/AF Frame.

Text can be entered into the comment fields using one or more of the following mechanisms:

- Typed in from the keyboard
- Pasted in via the clipboard
- Imported from a MS Word document

It is then saved to a SAS catalog SLIST entry using the Screen Control Language (SCL) 'Savelist' command. The linkage between the SAS dataset and the SAS catalog entry is accomplished by creating a unique identifier that is generated and stored as an 8-character variable. The identifier is used as part of the catalog entry name into which the comments are stored. This value is also stored in the SAS dataset that contains all the other information

about a given task. Comments for each observation are stored in a catalog entry of type SLIST. This type of catalog entry was used so that SCL lists could easily be stored and retrieved using standard SCL functions.

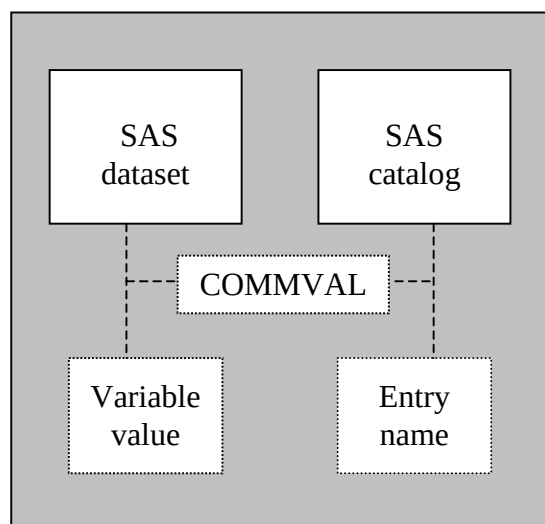


Figure 1

Details

This section describes some of the details of how SCL Lists are used to retrieve comments from SAS catalog entries, display the comments on the screen, and save them back to catalog entries.

We explored several different strategies on how to best implement the comment fields. The final implementation of the comment fields involved a two-tiered hierarchy of SCL Lists. Our goal was to be able to store all the comment fields for a given observation into a single catalog entry. We accomplished this by creating a 'master' SCL list and then populating it with multiple sub-lists. Each sub-list contained the comments for a single field. We used 'Named' Lists so that the name of each sub-list corresponded directly to the name of each comment field (which was an 'Extended Text Entry' widget on the AF screen). Another advantage of having all the comments for an observation stored in a single catalog entry was that we could programmatically assign a descriptive label to the catalog entry. This allowed the user or analyst to

quickly look at the catalog and determine if any comments had been added for a particular task.

Another important feature of this system involved designing a method for linking the exposure data, which were stored in a SAS dataset, with the comments that were stored in a SAS catalog entry. The objective was to create a simple, but permanent, link between each observation in the SAS dataset and its corresponding comment fields in the catalog entry. To accomplish this, we created an 8-character field that was saved as a variable in the SAS dataset and was also used as the entry name in the SAS catalog (see Figure 1).

This variable was constructed using the algorithm below.

```
commval='C' || left(put(ranuni(0)*10000000,7.));
```

This value was then used as part of the catalog entry name using code similar to the line below.

```
outslst='library.tasks.' || commval || '.slist';
```

We used the following SCL command to save the comments to a particular catalog entry.

```
rc=savelist('catalog',outslst,mstrlist,0,desc);
```

If the catalog entry was new, it would be created automatically, otherwise it would be overwritten. Appendix I shows the source code in our exposure estimation system that is used to save comment fields to a catalog entry. Appendix II shows the source code used to retrieve previously stored comments.

Importing text directly from a Microsoft Word document involved several steps. First, the user clicked on a small icon next to the comment field and was presented with a drop-down list box from which they could pick a document to import. The next step involved opening the selected document programmatically, selecting all the text, and then copying it to the clipboard. The last step was a simple matter of copying the text on the clipboard into the appropriate comment field.

The Master List and its children

Any observation which has comment fields attached to it has a corresponding entry in a SAS catalog. Each catalog entry has a single 'master' list (parent) which is populated with one or more sub-lists (children).

The figure below shows a conceptual view of how a single catalog entry is constructed using a list with multiple sub-lists.

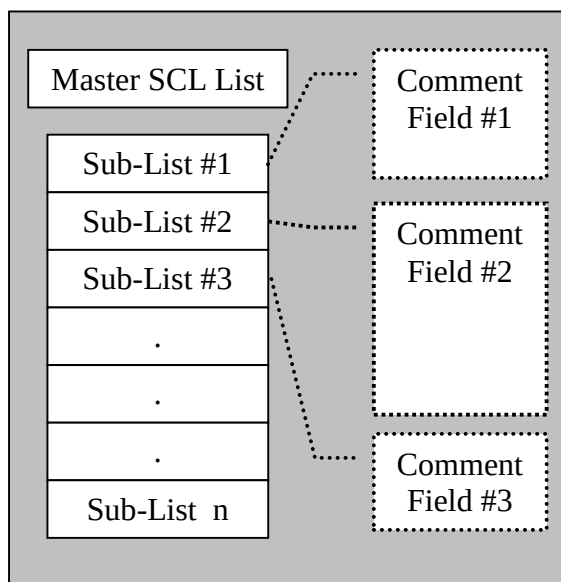


Figure 2

Conclusion

One of the primary reasons we chose to use SCL Lists for this system is their versatility. There are over forty SCL functions available in version 6 that can be used with Lists. Also, Lists can have sub-lists and they can be created, expanded, and deleted dynamically. Other features are that they can be used similar to arrays or stacks.

For all the advantages that we found in utilizing Extended Text Entry widgets as the base objects for our comment field subsystem, there was a

distinct disadvantage as well. The problem with using this particular type of object was that it could not handle special formatting codes embedded within the text. This meant that any text data that had been superscripted or subscripted or had other special formatting codes applied to it and was imported from an existing Word document or pasted in via the clipboard, lost the formatting information. Other than this deficiency, we have been extremely pleased with the versatility of SCL Lists used in conjunction with AF frame objects.

Trademarks

SAS and SAS/AF are registered trademarks or trademarks of SAS Institute, Inc. in the USA and other countries. ® indicates USA registration.

References

SAS Institute, Inc. (1996), *SAS/AF Frame Dictionary, Version 6, First Edition*, Cary, NC: SAS Institute, Inc.

SAS Institute, Inc. (1994), *SAS Screen Control Language, Version 6, Second Edition*, Cary, NC: SAS Institute, Inc.

Additional Information

For more documentation about how the comment fields work or information about the Exposure Estimation System, contact Robert Matthews at one of the following addresses:

Robert Matthews
University of Alabama at Birmingham
Department of Epidemiology
1665 University Blvd. Room 445
Birmingham, AL 35294-0022

E-mail: rsm@uab.edu

Web: <http://www.epi.soph.uab.edu/rsm>

APPENDIX I

Source code in our exposure estimation system that is used to save comment fields to a catalog entry.

```
array tmp1{*} tmp1-tmp6; * Array definitions would appear in the INIT section of the SCL program;
array comments{*} $8 com1-com6 ('SRCECOMM', 'MONCOMM', 'TSKCOMM', 'CATCOMM', 'EXPCOMM', 'TWACOMM');
* The names listed in the 'Comments' array are the widget names for each of the comment fields.
```

SAVECOMM:

```
  * Create a master list that contains sub-lists of all the comment fields.
  mstrlist = makelist();

  do i=1 to dim(comments);
    tmp1{i} = makelist();
    call notify(comments{i}, '_get_value_', tmp1{i});
    valid = listlen(tmp1{i});
    if valid > 0 then rc = insertl(mstrlist, tmp1{i}, 1, comments{i});
  end;

  *****
  * Check to see if any lists have been inserted into the master *
  * list. If so, then save the master list to a catalog entry. *
  *****
  valid = listlen(mstrlist);
  if valid > 0 then
    do;
      if commval = '' then commval = 'C' || left(put(ranuni(0)*10000000,7.));

      outslst = 'library.' || tsk_dsn || '.' || commval || '.slist';
      desc = 'PLT:' || plantnum || ' TASK:' || tasknum || ' MON: ';
      desc = desc || put(monnum,mon_fmt.) || ' CALYR1:' || calyr1;
      rc = savelist('catalog', outslst, mstrlist, 0, desc);
    end;

  do i=1 to dim(tmp1);
    rc = dellist(tmp1{i});
  end;
  rc = dellist(mstrlist);
RETURN;
```

APPENDIX II

Source code in our exposure estimation system that is used to retrieve previously saved comment fields from a catalog entry.

GETCOMM:

```
  incat = 'library.' || tsk_dsn || '.' || commval || '.slist';

  mstrlist = makelist(); templist = makelist();

  do i=1 to dim(comments);
    call notify(comments{i}, '_clear_');
  end;

  if cexist(incat) = 1 then
    do;
      rc = filllist('catalog', incat, mstrlist);

      * Go through the list and fill each comment field;
      do i=1 to dim(comments);
        found = nameditem(mstrlist, comments{i});
        if found then
          do;
            templist = getiteml(mstrlist,found);
            call notify(comments{i}, '_set_value_', templist);
            rc = clearlist(templist);
          end;
        end;
      end;

      rc = dellist(templist);
      rc = dellist(mstrlist);
    end;
  RETURN;
```