Paper 45-27

SAS/AF®: It's Not What It Used to Be!

Soora Wi, Kaiser Permanente, Oakland, CA

ABSTRACT

SAS/AF® Version 8 is a powerful application development tool that allows you to create a graphic user interface (GUI) to transform information into knowledge. This paper will introduce you, the first-time SAS/AF® (Version 8, Windows NT) user, application developer concepts to create GUI's for the non-SAS user, allowing them to "point and click" to view, query and print reports. This paper also provides steps to create an actual application. Throughout this paper, key SAS/AF® terms such as frame, components, attributes and SCL will be introduced.

INTRODUCTION

Today, access to data and the analysis of information is critical. In health care, this is no exception. At Kaiser Permanente, the legacy system contains millions of records in an array of databases, formats and platforms. At the Division of Research, Kaiser Permanente, researchers and analysts are dependent upon these systems to provide clinical data to investigate epidemiological and health services questions. These questions are supported by grants through various funding agencies such as the National Institutes of Health.

In order to obtain funds, researchers need to apply for grants provided by the government, or non-profit organizations. Although analysts are capable of extracting and reporting the data from these systems for grant proposals, the time required to complete a task may take hours or days. So, how can these researchers (non-SAS programmers) capitalize on the legacy data without repeatedly requesting extracts and analyses?

One solution is to develop a SAS/AF® application that will provide a GUI, allowing the researcher to submit a query to a SAS data table. The following sections will introduce you to key concepts towards developing this solution. Following the introduction and some brief definitions, step-by-step examples to create SAS/AF® frames with various components, attributes and methods are given to create an application.

WHAT IS SAS/AF®?

SAS/AF® is a SAS System application development tool that allows you to create interactive windows applications for the end-user. The interactive windows or GUIs are created using SAS/AF® Frames. These frames consist of components, which are subsequently controlled by various attributes, methods or SCL (SAS component language). All the entities that you create constitute an SAS/AF® application, which is stored in a container or catalog that you specify.

KEY SAS/AF® DEFINITIONS AND TERMINOLOGY

Library: A folder or subdirectory containing many SAS entities such as data tables, outputs and catalogs.

Catalogs: A container or storage area that can hold many SAS entities such as frames, lists, output, formats, SCL pmenus and keys. Catalogs are usually part of a library. SAS/AF® applications (all components) are typically stored in a named catalog. Catalogs can only be managed and accessed with SAS. Therefore, catalogs cannot be seen using MS Explorer®.

Data sets or tables: These terms are synonymous and will be used interchangeably throughout this paper. These are SAS datasets and tables.

Frame: A frame is the blank palette or screen, which will be enhanced with components. All components, attributes, methods or SCL added to a frame are always associated with that particular frame.

Component: A component is the graphical object, such as, buttons, entry fields and graphical text that is placed and sized onto the frame. A frame can contain multiple components.

Attribute: An attribute controls its associated component with specific qualities or characteristics. For example, attributes control the appearance and behavior of the component.

Method: A method is a specific type of SAS/AF® class that executes an operation on a component. Methods are typically invoked using dot notation.

Dot notation: A form of syntactical programming with SCL. It is called dot notation since a dot or period separates the name of a component and its attribute name.

SCL (SAS Component Language): SCL is an object oriented programming language. SCL can assist with frame branching or submitting SAS programs. A set of SCL code to control a frame is stored in the application catalog as a SCL entry. The name of this SCL entry does not have to match the associated frame.

DATA MART DEFINITION

The sample application in this paper is based on a table that will simulate a data mart. By definition, a data warehouse or data mart is "the extraction, cleansing and transformation of operational data for the purpose of decision support". It has also been defined as "a container that stores information of interest such as ad hoc queries or cross-subject analyses to a specific department, individual or area of interest".

Similarly, aggregate data is information that is summarized by a few or several variables. Aggregate data usually is stored at the level of the event. In this sample health research application, the aggregate data represent the number of explicit diagnoses, grouped by several variables.

APPLICATION DEVELOPMENT STRATEGY

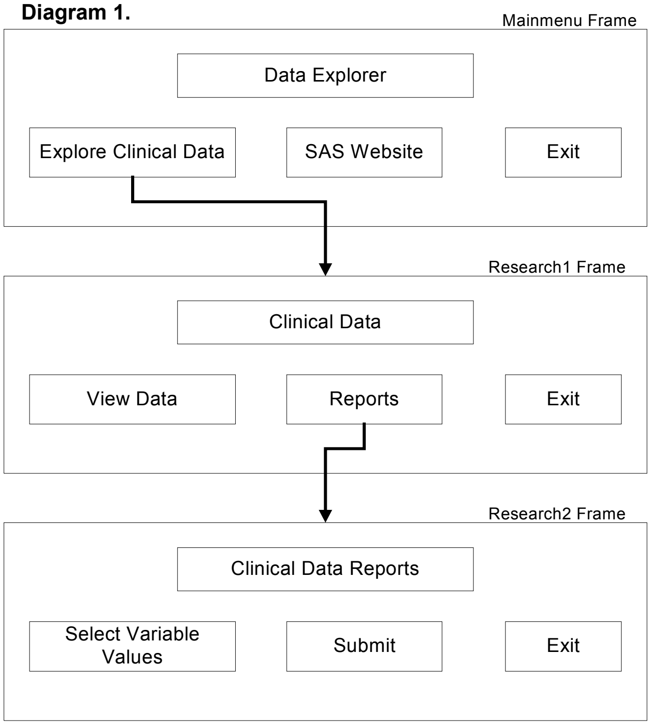
In creating a SAS/AF® application, here are some things to consider before starting.

1. Who are your clients/users of the application?
2. What platform is SAS running?
3. How many clients/users will use the application?
4. Will the application be on a stand-alone PC or will it reside on a server/LAN PC?
5. Are you prepared to undertake an iterative process in creating a SAS/AF® application?

If you have readily addressed the questions above, you are strongly encouraged to envision the entire project/application as it's being created. This can be accomplished clearly (to you and the user) by creating a flow diagram (Diagram 1) of the processes in the application.

Once the diagram is created, use it as a check for each completed section. Your diagram could be as simple as outlining the major processes in the application or can be as detailed as possible, showing each frame or component interaction.

Diagram 1.



DATA MARTS WITH PROC SUMMARY

The following sample health research SAS/AF® application will be designed with a very simple data mart in mind. Since you are creating the application, you must be very familiar with creating this “back-end” table and all of its contents, including all variable names and their possible values.

In this example, the data mart is created using PROC Summary on the mainframe. To create an aggregated dataset describing hospitalizations for a given year, submit the following code:

```
proc summary data=onerec;
  class age race sex diagnosis;
  output out=summary
run;
```

Since the application will be stored on a LAN (local area network), the aggregated dataset will be downloaded to the LAN by FTPing a transport file. This process involves creating a transport file using PROC cport and using PROC cimport after the download process.

CREATING A NEW LIBRARY AND CATALOG

The following section will guide you through the creation of a new library and catalog. To create a new library and catalog follow these steps:

1. Examine the left window that contains the SAS Explorer window.
2. Right mouse click (RMC) on Libraries.
3. Select New.
4. Enter a name for your new library, e.g. KP.
5. After the library called KP has been created, double-click on Libraries.
6. RMC on the library named KP and select New.
7. Double-click on Catalog and name the catalog, e.g. Research.

From here, all following frames will be contained in the library called KP and the catalog named Research. Reference to these names will be made at the end of the paper in order to create a shortcut to execute the application.

CREATING FRAMES

The following sections are the examples of creating three frames, Mainmenu, Research1 and Research2, with various components, attributes and methods. Following Diagram 1, this Mainmenu will give the user major categories to select. Each selection should enable the user to cascade into additional selections and processes. The first frame we will create is the Mainmenu frame.

1. RMC on the catalog called Research.
2. Select New and double-click on Frame.
3. A blank screen or frame will appear and will currently be untitled until saved. This is the SAS/AF® build mode. At this point, a list of available components will appear in a window on the left.
4. Select Graphic Text Entry Control by click, drag and drop (CDD) onto the untitled frame.
5. Open the Properties of the Graphic Text Control object and enter the following attributes:

Attribute Name	**Value**
name	Title
text	Data Explorer
textColor	black

You can find other attributes for this Graphic Text Control component by double-clicking on the component and selecting attributes and any subsequent parameters such as borderStyle, backgroundColor and font. To review, the steps above created a frame called Mainmenu. On the Mainmenu, a Graphic Text Control component with an Attribute Name Value of Title was created.

This Mainmenu frame will be the initial frame displayed when the application is executed. For the user to exit the application, place a button on the Mainmenu frame so that once selected, the application stops and exits SAS. The following steps create this button.

1. Since this exit button will be on the Mainmenu, open the Mainmenu frame and select Push Button Control from the components window.
2. Click, drag and drop (CDD) the Push Button Control onto the Mainmenu frame.
3. Change the value of each attribute name to the following:

Attribute Name	**Value**
buttonStyle	Icon with text under
iconStyle	Large icons
label	EXIT
commandOnClick	end
name	EndAF
toolTipText	Exit Data Explorer
icon	111 (A variety of icons are available, with each icon having an associated numeric value.)

To review, a new Push Button Control component was added to the Mainmenu frame. The name of this Push Button Control component is EndAF and appears with a large icon with the text, Exit Data Explorer underneath the icon.

A useful function from the Mainmenu frame is to have the user be able to link to a web site. The following steps allow the user to create such a link on the Mainmenu frame.

1. Rather than selecting another Push Button Control component from the component window and since this button is similar to the Push Button Control just created, simply cut and paste the previous Push Button Control onto a different location on the Mainmenu frame.
2. Change the value for each of the following attribute names:

Attribute Name	**Value**
buttonStyle	Icon with text under
iconStyle	Large icons Label
label	SAS Website
commandOnClick	wbrowse "http://www.sas.com/"
name	Weblink
toolTipText	Link to SAS Website

1. Additionally changes to the iconStyle as well as many other attributes can be made.

To review this section, a Push Button Control component named Weblink was created and added to the Mainmenu frame. This Push Button Control executes a command called wbrowse to initiate a web browser to open the SAS web site. After closing the web site, the SAS/AF® application will return to the Mainmenu frame.

The final component on the Mainmenu frame will be a Push Button Control to enable additional selections for utilizing the data mart that was created earlier.

1. Since this button will be on the Mainmenu, open the Mainmenu frame and select Push Button Control from the components window.
2. Click, drag and drop (CDD) the Push Button Control onto the Mainmenu frame.
3. Change the value of each attribute name to the following:

Or

1. Copy and paste the Push Button Control component named EndAF onto the Mainmenu frame.

Attribute Name	**Value**
buttonStyle	Icon with text under
iconStyle	Large icons
label	Explore Clinical Data
name	ClinicalData
toolTipText	View, select, print data

Rather than issuing a commandOnClick attribute to this component, this component will be controlled by a few lines of SCL, SAS Component Language. However, before introducing SCL and dot notation syntax, additional frames should be built, representing "targets" from the Push Button Control labeled Explore Clinical Data.

1. Add a new frame in the catalog called Research1.
2. Copy and paste the Graphic Text Control component labeled Data Explorer from the Mainmenu to the new frame and label it as Clinical Data.

On the frame labeled Research1, three components will be added. The first component is another Exit Push Button Control for the user to exit to the previous menu.

1. Copy a previous Exit Push Button Control.

The second component will be a Table Viewer Control.

1. CDD a Table Viewer Control onto the frame from the components menu.
2. CDD a SAS Data Set Model from the components window. Make sure to place the SAS Data Set Model directly onto the table viewer control. This establishes a model/view relationship.
3. Change the properties of the Table Viewer Model to the following:

For the Table View Control:

Attribute Name	**Value**
label	ViewTable
model	Sasdataset1

For the SAS Dataset Model:

Attribute Name	**Value**
description	SAS Dataset Model
name	Sasdataset1
table	<libname.datasetname>

The library name will be established in the following SCL.

USING SAS COMPONENT LANGUAGE (SCL)

The third component to place on the frame will be a Push Button Control labeled Reports.

1. CDD a Push Button Control onto the Research1 frame.
2. Open a SCL for the frame.
3. Write SCL to call the next frame called Research2.

To control the flow from the Mainmenu frame to other frames, such as Research1, SCL code must be written. While building the Mainmenu frame, the following SCL must be created:

```
INIT:

rc=libname('dsn','i:\home\dream\dea\saslib');
    put rc;
  msg=sysmsg();
    put msg;

return;

CLINICALDATA:
  call display('research1.frame');
return;
```

Here, the SCL establishes a library name where the SAS table exists and controls the flow of the Push Button Control named ClinicalData to the Research1 frame.

To review, a second frame was created with several components. A Table Viewer Control was introduced to allow the user to directly view a SAS data set. Additionally, a Push Button Control to take the user to another frame was created.

The final frame for this sample application will be called Research2. This frame will be initiated from the Push Button Control labeled Reports from the previous frame. The purpose of this frame is to allow the user to select values from a variable for analyses. In this frame, there will be four components. They will consist of an Execute and Exit Push Button Controls, a Radio Box Control and a Graphic Text Entry Control.

For the Exit Push Button Control, simply copy one from another frame. However, modify the new Exit Push Button Control with the following:

Attribute Name	**Value**
commandOnClick	clear output; end;

This Exit push button will clear the output from a generated report and return the application to the previous frame

For the Graphic Text Control, copy the Graphic Text Control from the previous frame and modify the attributes to:

Attribute Name	**Value**
text	Clinical Data Reports

For the Radio Box Control, select properties and modify the attributes to the following:

Attribute name	**Value**
borderTitle	Select Diagnosis Code
defaultAttribute	selecteditem
name	selectdrg
items	… (Change the default Radio Box Control values to known values of the selecteditem).

Finally, add a push button onto the frame with the following attributes:

Attribute name	**Value**
buttonStyle	icon with text to right
icon	296
iconStyle	small icons
label	Execute

To control the flow from the Research1 frame to the Research2 frame, the following SCL can be used:

```
Reports:
  call display('Research2.frame');
return;
```

In order to control the value selected in the radio box control in the Research2 frame, SCL must be written. For example, to execute a SAS program or submit block that uses the value selected in the radio box control the following SCL must be written:

```
dcl num drg;

Execute:

  drg=SelectDrg.SelectedItem;

submit continue;

 call execcmdi('list on');

 data temp1;
  set dsn.clindata;
   if diagnosis=&drg then output;
run;
title 'Frequency of Diagnosis=&drg, by Sex
and Age Group';
proc freq data=temp1;
  tables sex*age;
run;
endsubmit;

return;
```

To review, the final frame was created that allows the user to select values of a particular variable and execute a submit block. The communication between frames, components and data sets are accomplished by using SCL. Upon executing the final frame, the report is displayed in the output window.

CREATING A SAS/AF® SHORT-CUT FOR THE DESKTOP

One method to automate the SAS/AF® application just created is to create a desktop icon that will initiate the application. You can achieve this by the following steps:

1. Copy the SAS icon and rename it Data Explorer.
2. RMC the new icon and select properties.
3. Select target.
4. Type the following options:

```
-initcmd “af c=kp.research.mainmenu.frame”
```

Enhancements to this short-cut command can be added. Here are some other possible commands to add. Please remember that these are commands that are separated by a semicolon.

Additional commands	**Definition**
Toolclose;	Closes the toolbar
Commandclose;	Closes the command line
Wwindowbar off;	Turns off the window bar
-awsdef 0 0 100 100	Opens to full screen
-awscontrol nosystemmenu	Hides pmenus
-awstitle “Data Explorer”	Border title
-splashloc <filename.bmp)	Shows a splash screen (.bmp file only) before the application opens.

CONCLUSION

This paper introduced you to some key concepts in developing a GUI to access and analyze data using SAS/AF®. The information provided is only a small portion of the numerous tools that are available using SAS/AF®. Further experience with SAS/AF® will enable you to create applications that automate many tasks. As a developer, you can utilize the power of SAS/AF® to create and maintain tracking systems and other detailed data management and analytical tasks.

ACKNOWLEDGMENTS

The author would to express his appreciation to the following for their assistance with this paper:

Steve Wilson, MAJARO InfoSystems, Inc

The technical support staff at the SAS Institute.

CONTACT INFORMATION

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

Division of Research, Kaiser Permanente
3505 Broadway
Oakland, CA 92611
Work Phone: 510-450-2334
Fax: 510-450-2072
E-mail Address: sxw@dor.kaiser.org