EMPOWERING RESEARCHERS: USING SAS/AF TO CREATE A GRAPHICAL USER INTERFACE TO SAS, SYBASE, AND THE UNIX SYSTEM

Charles Perdue Sutelan
Federal Judicial Center
Research Division

INTRODUCTION
The Federal Judicial Center's (FJC) Information Access System represents an attempt to create a graphical user interface to SAS, Sybase, and the UNIX System using SAS/AF and SAS/EIS. This system was designed to provide legal researchers with the ability to function in a complex computing environment. Having recently converted from a VAX/VMS operating system using System 1032 as the DBMS to a UNIX platform running Sybase, we faced the difficult problem of developing a user-friendly interface to both the FIC's Integrated Data Base (large database containing 25 years of historical data from the federal court system) and project-specific data. Essentially, we were looking for a degree of flexibility which would allow users to perform, on an ad hoc basis, the kind of operations traditionally performed by information systems personnel. Ideally, this interface would allow users to not only generate reports, but to create and load Sybase tables, generate large subsets of data, and dump data in a format amenable to transfer to other platforms and applications. In addition, it was desirable to create a "desktop" interface to the UNIX system itself, providing users with a point-and-click environment from which to run various applications including SAS/Assist, Sybase's Data Work Bench, SPSS, and a basic text editor, NEDIT.

Thus, the guiding principal in the development of this interface was flexibility. We needed an approach which would empower users with a whole host of utilities and applications not specifically linked to any particular dataset. The major purpose of these utilities was to make each user a potential manager of his own database.

This paper is divided into two basic parts. The first section describes the structure of this application with a special emphasis on the interrelationships between the different utilities and windows. The second section provides specific details on the construction of several of the Sybase utility and data extraction screens, offering examples of SCL code and figures illustrating the layout of these windows.

I. DESCRIPTION OF THE APPLICATION
This application is comprised of a series of nested windows emanating from a main menu composed of eight icons. These icons provide the user with the ability to invoke applications, extract DBMS data, review federal court directory information, invoke project-specific applications, and perform a number of client-server functions. The layout of the main menu and the structure of the system is illustrated in Figure 2.

The main menu consists of a SAS/AF Frame entry which is opened upon invocation of the SAS system by a line in the user's "autoexec.sas". The design of this window is intended to suggest a control panel from which users may branch off into various applications. The first three icons on the top row launch, respectively, Sybase's Data Work Bench, SAS/Assist, and SPSS's display manager. Clicking on any of these three icons simply uses the "X" command to send to the UNIX system the text of the word necessary to launch the application (See Figure 1). The fourth icon opens a custom designed interface to the FIC's historical database located in Sybase. Using this interface, users can create SAS data sets or views containing selectively defined subsets of IDB data. More specifically, users can select the type of data (Civil, Criminal, Appeals, or Trials), the cohort (terminated or filing), the year specification, and the particular variables desired. These subsets, or "data marts", are then available to the enduser either for decision support applications or for online analytical processing (more information on this window is provided in Section II of this paper).

![Figure 1: Command Processing Window for Invoking Data Work Bench](image-url)

The icon entitled "Court Directory" consists of a series of windows displaying information on federal court personnel. From these windows, the users can search for individuals and groups using a variety of criteria as well as produce mailing labels. The data for this application are physically located in Sybase and are made available to SAS via a view descriptor created specifically for this application.
Figure 2: Federal Judicial Center's Information Access System: Main Menu
The project button displays the names of current division projects for which specialized applications have been created. The items in this list are stored in a SAS dataset and are displayed dynamically using the `datalistc` function. In addition to the application name, this dataset also contains the SAS library, catalogue, object name, and object type of the window to be displayed when the application is selected. Thus, each record in this dataset contains an application for which an information system has been constructed as well as the information necessary to invoke it. In addition to a 'project names' dataset, a 'user authorization' dataset is also maintained. This dataset contains a list of division staff with a series of fields identifying projects to which they have access. When the user selects the "Projects" icon, a choice list is displayed listing the available projects. A selection from this list, prompts the program to open the authorization dataset, find the name of the current user in that list and, using an array, check if the selected project name appears in any of the fields. If the project name is found, the project's associated application is invoked using the catalog information located in the project names dataset. If the project name is not found, the user is denied access and an appropriate message is displayed. The user's name, which is used in this comparison, is extracted from the UNIX environment variable, "$USER", using the `sysget` function in a submit block. Figure 3 illustrates the labeled block associated with the "Projects" icon.

The "Utilities" icon invokes a selection list which displays a series of functions or screens. These functions comprise both client-server and UNIX system utilities. The client-server features include screens for creating SAS access descriptors of Sybase tables, screens for creating Sybase tables from SAS datasets, and a window which allows the user to dump either SAS or Sybase data into a raw, tab-delimited ASCII file.

Operating system functions include the ability to open an xterm window from which to issue UNIX commands, and the ability to invoke a UNIX text editor. Additionally, the list includes a selection which opens and activates the SAS program, output, and log display manager windows. [Note: To "clean up" the initial display, both the log and the program window are blanked out when the main menu is invoked using the "log off" and "program off" commands] and a selection which activates SAS's query window.

The last icon on the main menu closes the window and terminates the current SAS session.

These eight icons are intended to provide the end-user with a graphical interface to the Judicial Center's computing system. While the system is constructed using SAS, that fact should be transparent to the user who, using the main menu, has the option to select the processing software of her choice (Data Work Bench, SAS, or SPSS). In addition, the utility for dumping SAS/Sybase datasets into tab-delimited ascii files makes the data available to such Macintosh based applications as Excel, Filemaker, Word, Statview, etc.

```
PROJECT:
prds = open('sasuser.projnm');
cntlst = makelist();
rc = curlist(cntlst);
project = datalistc(prds, 'project', 'Select a project');
N = getnitemn(cntlst, 'COUNT');
if N > 0 then do;
   rc = locatec(prds,l,proj);
   lib = getvarc(prds,2);
   cat = getvarc(prds,3);
   name= getvarc(prds,4);
   type= getvarc(prds,5);
   location =
      compress (lib||"."||cat||"."||name
             ||"."||type);
rc = close(prds);

   dsid=open("sasuser.names(where(user='"||
         upcase(name)||"'))");
   Do i = 2 to 11;
      rc = locatec(dsid,i,upcase(proj));
   if rc ne 0 then do;
      call execmd('afa c='||location);
   end;
   else _msg_ = 'You are not authorized
to access this Project: See System Administrator';
   end;
   rc = close(dsid);
end;
RETURN;
```

**Figure 3: SCL Code Used to Verify Project Access**

### II. TECHNICAL DETAILS

**SYBASE DATA EXTRACTION PROGRAM**

In an effort to facilitate user-initiated research, a screen was created which would allow users to create their own subsets of the Judicial Center's historical database. These data are physically located in Sybase in three tables, the first of which contains cases terminated between SY70 (statistical year) and SY87 in the federal courts. The second table contains those cases which were terminated between SY88 and SY94 and the third table contains those cases which were still pending in the federal courts as of October 1, 1994.
This utility application is comprised of only one SAS/AF FRAME entry screen. The layout for this screen is relatively simple, consisting of a series of push button classes and a single text entry field (See Figure 4). The push buttons are divided into three groups dependent on their function in the application. The first four push buttons, under the heading of "Data Types", allow the user to select what type of cases (Civil, Criminal, Appeals, or Trials) to subset. Upon selection, the choice is confirmed visually with a message indicating the type of cases selected. In addition, the SCL program uses this selection to set the associated SAS/Access view descriptor names.

The next button invokes a pop-up menu which allows the user to either select all fields or to specify which fields she would like included in the subset. If the user selects the latter choice, a list of the available variables is displayed using the varlist SCL function to produce a pop-up menu containing the fields in the view descriptor associated with the type of data being processed (i.e., if the user selected "Civil" as the type of IDB data, then the varlist function is passed the name of one of the view descriptors accessing the civil data).

The last group of buttons, entitled "Actions", consists of two push button objects allowing the user to either create a SAS dataset or a SAS view using the previous specifications. Clicking on either of these buttons submits a SQL command to the SAS processor. Depending on the specifications, this command may create a SAS dataset or a SAS view.

---

1 SAS/Access View descriptors are files containing a complete description of Sybase tables including the Sybase table name, DBMS column names, SAS variable names, and any relevant subsetting information.
Interactive Systems

a dataset/view from a single Sybase table or may combine up to three tables using the "outer union corresponding" feature of SQL (See Figure 6).

CRVIEW:
If yr beg le 87 and yr end le 87 then
do;
submit continue sql;
create view &libname as
select &vars
from &type
&whecl;
endsubmit;
end;
else if yr beg le 87 and yr end ge 88
then do;
submit continue sql;
create view &libname as
select &vars
from &type1
&whecl
outer union corresponding
select &vars
from &type2
&whecl;
endsubmit;
RETURN;

Figure 6: Code Used to Create SAS Views of Sybase Tables

SAS/ACCESS DESCRIPTOR PROGRAM
The major function of this window is to provide users with the ability to establish their own link between the SAS system and Sybase tables using SAS/Access procedures.

This application consists of a SAS/AF Frame entry window containing two list boxes and several push button objects (See Figure 7). Upon invocation, the list box on the right side of the window displays a list of the current Sybase databases. A selection from this list causes the names of the tables in that database to be displayed in the second list box on the left side of the window. Upon making a selection in either list box, the contents are cleared and only the selected item is displayed, highlighted in red. Two buttons located between the tables allow the user to deselect the database/table and display the full list.

A third push button object invokes a dialog window which requests the user to enter her Sybase account name, Sybase password, and the name of the resulting SAS access descriptor. A checkbox located under this button allows the user to indicate whether or not the program should assign names to the fields.

Lastly, three push buttons located on the bottom of the screen allow the user to select a SAS library in which to store the descriptor, create the new access descriptor, and invoke the Access window in which she may edit the descriptor or create a SAS view.

Figure 7: Access Descriptor Screen

In order to display the list of Sybase databases and dynamically create a list of tables in those databases, this program makes use of the SQL pass-through facility. This process combines the SQL procedures available in the base SAS software with the access features of SAS/Access to retrieve data directly from Sybase tables. Unlike data retrieved via SAS/Access descriptor files, the pass-through procedure represents a more efficient use of the client-server architecture because it allows the SQL server in Sybase to optimize the query by taking advantage of indexed columns. More specifically, pass through queries pass a Transact SQL query directly to Sybase where this query is processed. Upon completion, only the results of the query are transferred to SAS where the data may be displayed or subjected to further processing. This utility is especially important when processing queries containing such aggregate functions as "Count", "Average", and "Sum".

The present program uses the SQL pass-through procedure to create SAS/views of the Sybase system tables containing a list of the current databases and tables. When this screen is opened, a submit statement in the initialization phase runs a SQL process which establishes a connection to Sybase and creates a view of the "syssystems" table in the database "Master", eliminating from the view databases used by the system (i.e. those databases beginning with the prefix "syb") (See Figure 8).

A selection from the list box containing the Sybase databases causes a second submit block to be processed. This block incorporates the name of the selected database...
to produce a SAS view of the current, non-system, tables located in that database (Note: each database contains a list of objects in that database in the system table "sysobjects": tables can be identified by their type('U')).

```
Submit continue sql;
connect to sybase as mydb (Server=svname database=master user=usname
password=XXXXXX);
create view work.dbtempv as
  select name from connection to mydb
    (select name from sysdatabases
     where name not like 'syb%');
disconnect from mydb; quit;
endsubmit;
```

Figure 8: Submit Block for Creating a View Containing Current Sybase Databases

The view created by this block of code is then used to repopulate the second list box displaying a list of the tables in that database. (See Figure 9).

```
DATABASES:
  length dbname $ 30;
call notify('dbases', '_SELECTED_', 1, row);
if row > 0 then call notify('dbases', '_GET_TEXT_', row, dbname);
if dbname ne '' then
  do;
  Submit continue sql;
  connect to sybase as mydb (Server=svname database=dbname
user=usname password=XXXXXX);
create view work.tables as
  select name as tbname from connection to mydb
    (select name, uid from sysobjects
     where type = 'U' and name not like 'syb%');
disconnect from mydb;
quit;
endsubmit;
call notify('tables', '_REPOPULATE_');
call notify('dbases', '_DELETE_ALL_');
call notify('dbases', '_ADD_', dbname, 1);
call notify('dbases', '_SET_COLOR_', '', 'red');
call notify('tables', '_GET_MAXROW_', numrows);
if numrows = 0 then _msg_ = 'Operation Failed: You May Not Have Permission to Access These Tables';
end;
Return;
```

Figure 9: SCL Code for Databases List Box

After selecting a Sybase database and table name and entering their identifying information, the actual access descriptor may be created by selecting the "Create" button at the bottom of the window. If the user has failed to enter all of the necessary information, the program is not run and a warning message is displayed requesting the user to supply the requisite data. Otherwise, a submit block is processed which creates a SAS/Access descriptor file of the Sybase table. The program then verifies that the submit block was successful using the sci exist function to check if the access descriptor was created. If the user does not have permission to access a particular table or database, then the function will return a "0" in which case an error message is printed warning the user of this situation and the "tables" list box is left empty.

DATA REVIEW AND ANALYSIS SCREEN

In this final section, I will discuss specific details of one of the project-associated analysis screens built using SAS/AF Frame. This screen is accessed through the "Projects" button on the main menu and is only available to those users working on the project. The layout of the window itself is fairly generic and is used for a number of different projects with minor modifications (See Figure 10). These modifications include changes to the title of the window as well as the objects included in the "Report Options" container box located in the bottom right-hand corner of the screen.

```
CJRA Demonstration Project

Survey Data Review and Analysis

Survey Fields
  Input Variables
    Domain Variables
      Demographic
      Other Variables
  Output Variables
    Codebook
    Demographics
    Other Variables
  Output Formats
    Text
    Excel
  Output Options
    Report Options

Present Reports
  Input Reports
    Codebook
    Demographics
    Other Variables
  Output Reports
    Text
    Excel
```

Figure 10: Project Review and Analysis Screen

Basically, this screen provides the user with a quick and efficient means of extracting information from a SAS dataset (or Sybase table using a view descriptor). From this screen, the user can produce one-way frequency distributions, cross tabulations, and univariate statistics as well as dynamically graph the data in an associated screen. In this way, this window is not unlike the frequency.
univariate, and graph screens available through SAS/Assist. However, unlike those windows, the Data Review and Analysis Screens are specifically designed for the datasets to which they provide access including specialized subsetting features. Two list boxes on the screen display a list of variables in the dataset and a list of preprogrammed reports. The report options container box contains several radio buttons with which the user can produce where clauses and apply these restrictions to the reports. Each of these radio buttons is linked to a variable in the dataset, usually a demographic or categorical variable that the analyst might logically use to subset the data. In addition, the user has the option of invoking an additional window in which she may generate a customized where clause (See Figure 5).

One of the most unique features of this window is that it allows the user to either print the output (frequency or univariate) to the output window or display it in a special pop-up window. Basically, two push buttons are provided for each procedure the screen performs (frequencies and univariates). The first button, entitled "Quick Frequencies" [or "Quick Univariate"], submits a block of code which runs the chosen procedure on a variable selected in the list box and saves the output to an SAS data file. In the same step, those data are displayed on the datalistc function. For frequencies, this function retrieves the screen the formatted values of the selected variable as well as the frequency and percentage associated with each value. A header is provided for the window by using the datalistc function. For data generated by the univariate procedure, the header is present in the function to denote the statistics being displayed. The procedure produces values representing the N, minimum, maximum, mean, and median.

The second type of function provided for each procedure, initiated through the button entitled "Output Frequencies" [or "Output Univariate"], performs the same basic operation with the exception that all output produced is sent to the Output window. Two push button objects located on the bottom right of the main window allow the user to clear this window or print the output. These commands are executed using the woutput scl function.

An additional option is available with the "Output Frequencies" push button which provides the user with the ability to cross tabulate the variables. Upon selecting this button (if the user has selected two variables from the selection list), a pop-up menu is invoked which asks the user if she wishes to produce a crosstab to which the user may select either 'yes' or 'no' from the list provided. If the user responds in the affirmative, the procedure prints both a one-way distribution as well as a cross tabulation of the variables.

In order to prevent errors, the univariate procedure uses the vartype function to insure that the variables passed to the procedure are, in fact, numeric. Unlike the frequency button which accepts both character or numeric, this procedure only works for the latter type of variables.

```sas
QUICKFRQ:
If varl ne ' ' then do;
submit continue;
proc freq data = sasuser.cjramang
noprint;
tables &varl / out = freq;
&whrcls
&where
run;
endsubmit;
vardisp = varll||" count percent";
dsid = open('work.freq');
label =
varlabel(dsid,varnum(dsid,varl));
call wregion(10,10,15,70);
dstype = open(sasuser.cjramang');
type =
vartype(dstype,varnum(dstype,varl));
rc = close(dstype);
if type = 'N' then
nothing =
datalistn(dsid,vardisp,Label,'N',1);
else if type = 'C' then
nothing =
datalistc(dsid,vardisp,Label,'N',1);
rc = close(dsid);
end;
else _msg_ = 'Please Select a Variable from the Above List';
Return;
```

Figure 12: SCL Code Used to Generate and Display Frequency Distributions to the Screen

A push button object located at the bottom of the window invokes an additional window in which the user may produce graphs. The labeled section associated with this object contains a submit continue SQL statement which creates a SAS view in the users work library. This view contains all the variables in the original dataset and
the records which met whatever subsetting criteria had been specified in the analysis screen. The call display function is then used to invoke a secondary AF Frame window.

This window consists of a graphics object, two sets of radio buttons, and several push button objects (See Figure 13). The push buttons allow the user to select a response variable, a grouping variable, create titles, print the graph, and close the window. The radio buttons allow the user to toggle between types of graphs and different statistical options.

Upon invocation, no graph is displayed and a message is displayed requesting the user to select a variable to graph. Selecting the "Chart Variable" button opens a small menu displaying a list of all the variables in the view. A selection from this list produces a horizontal bar graph illustrating the frequency of the responses to that variable. The graph object defaults to a horizontal bar graph with the default dataset or view being "work.graph". Similarly, the "Group Variable" button displays a variable selection list, a choice from which groups the bars in the graphics object according to its values. The "statistic" radio button allows the user to toggle between displaying either the frequency or the percentage in the graph. In addition, the user may click on any bar in the graph to produce a small box containing the vital statistics of that value (this option is activated using the _show_click_info_scl frame function). The "Chart Type" radio button allows the user to select the type of chart to display. While there is no button with which to select a dependent variable for the chart, selecting "Scatter Plot" from the chart type radio button produces a list of numeric variables to designate as the dependent variable (See Figure 14). A push button entitled "Titles" opens a small program entry window which allows the user to enter a custom title to be applied to the graph. After entering a title and pressing the "Return" key, this window is removed and the new title is applied to the graph using the following command:

```
call notify('Graph', '_SET_TITLE_', 1, title);
```

Lastly, the "Print Graph" button sends the graph to the user's default network printer.

```
Figure 14: Labeled Block for the "Chart Type" Radio Button
```

Overall, these two windows (Analysis and Graph) provide the enduser with a powerful and intuitive means of extracting information from a SAS dataset or Sybase table. The most powerful feature of these windows is their flexibility and their amenability to rapid prototyping. In less than an hour, the analysis screen can be copied and modified for another project. Even more flexible, the graph screen can be used by any calling application which creates a view named "work.graph" with all control over subsetting of the original dataset being performed at the level of the calling program. The graph screen simply uses the resulting view to perform its functions. As a result of its non-project specific nature, the graphing window is maintained in an "utility" catalogue and is used by multiple project applications.

**SUMMARY**

In a research environment, the critical factor is versatility. An application which provides only preprogrammed reports has a limited utility for researchers performing their own ad hoc examinations of a database. Rather, the key lies in applications which are both simple to use and accept maximum user input through the medium of push.
buttons, selection lists, radio buttons, hot-spots, etc. In addition, these applications must be able to read and manipulate data in multiple locations, both in the native SAS format and in remote RDBMS tables. SAS/AF, in conjunction with SAS/Access to Sybase, proved an ideal software combination for constructing a powerful graphical user interface to meet the ever-changing needs of our division.

REFERENCES

