Army Hearing Evaluation Automated Registry System (HEARS) Corporate Data Reporting System -- A Customized EIS Solution

Krista Elspas, Troy Systems, Inc. -- Fort Detrick, Frederick, MD
Julie Shadoan, Dept. of the Army -- Fort Detrick, Frederick, MD

ABSTRACT

The Army HEARS Corporate Data Reporting System is a customized enterprise information system operating in a client/server environment, utilizing SAS/AF® Frame entries for its user interface. The system is designed to provide the Army Hearing Conservation Program managers and consultants with the ability to easily generate a series of standard and customized reports that utilize corporate data collected from all of the Army hearing conservation programs operating worldwide. Standard reports are generated using base SAS software on an MVS mainframe, and then stored on a Windows NT server for quick access locally. Custom report requests can also be submitted locally from within Windows 95 via SAS/CONNECT® for execution on the MVS mainframe using SAS Macros and DATA steps. When a custom report is completed, it can then be downloaded to the requester's PC for viewing and/or printing via the REPORT procedure. The ability to further analyze and manipulate the generated SAS data sets via SAS/ASSIST® and Microsoft Excel from within the reporting system greatly enhances the user's capabilities to quickly and efficiently evaluate the effectiveness of the Army Hearing Conservation Program from many different perspectives. This paper describes the development process and obstacles overcome to achieve this customized EIS solution.

INTRODUCTION

The Army HEARS Corporate Data Reporting System was developed for a small group of Hearing Conservation Program managers and consultants located at the U.S. Army Center for Health Promotion and Preventive Medicine (USACHPPM) at Aberdeen Proving Ground, Maryland. The original reporting application utilized FOCUS® for data storage and retrieval on an MVS mainframe located at Fort Detrick in Frederick, Maryland. The system's interface was developed in FOCUS and consisted of a structured set of navigational menus and report programs.

As the business processes and objectives of the Hearing Conservation Program managers and consultants changed and the amount of data collected began to exceed the storage limitations of FOCUS, the overall limitations of the Army-wide HEARS Reporting System became exceedingly obvious, and the need for more flexible, streamlined and robust reporting capabilities via a client/server application became evident. After attending numerous SAS user group conferences and SAS Institute marketing demonstrations, we determined that SAS could provide us with the many tools we would need to develop a customized EIS solution utilizing client/server technology.
Being new to the SAS System for PCs in a client/server environment, initial development was slow and sometimes very difficult. Our first attempt at menus consisted of using SAS/AF programs coded in SCL to create text entry screens. Report generation itself was accomplished by storing and parsing PROC REPORT output from the mainframe and then recreating the report on the PC. This approach became too rigid, so we requested on-site technical support services from SAS Consulting® to help us find other options. Two SAS consultants spent two days with us, during which time they provided us with a wealth of tips and techniques. As a result, we became familiar with SAS/AF Frame entries and based our report generation on SAS data sets rather than flat files. The journey towards the current version of the Army HEARS Corporate Data Reporting System was on its way.

MAINFRAME REPORTS

In our application, all report data used to produce reports on the PC are generated on the mainframe in batch jobs using DATA steps and various procedures including FREQUENCY, MEANS, SUMMARY, and SQL. To provide for the appropriate amount of workspace for the one million or more records processed by these programs, we used the REGION= option on the JCL job card and the MEMSIZE= option in the SAS options statement.

Standard reports are generated twice each year following an upload of new hearing conservation data. These reports utilize the same criteria each time they are run, and are based on the two most recent calendar years included in the current data upload. The resulting data are downloaded using SAS/CONNECT and stored on a Windows NT server for access via the Standard Reporting Menu.

Custom reports are submitted from the Custom Reporting Menu and run in batch mode on the mainframe. The resulting data sets are stored for twenty days in a report data library for access via the Custom Reporting Menu. However, custom reports are required to utilize user-specified criteria such as date ranges and Army installation zip codes. In order to support this requirement, the standard report programs were converted into SAS macros. To send the user-specified parameters to the macros, the parameters must first be defined in the %MACRO declaration statement. When referencing the parameters in later code, the parameter names must be preceded by an ampersand (e.g., *&start* or *&stop*).

Depending on the values of these parameters, certain sections of code may or may not be executed. A good example of this is in designating the type of data on which to base the report. If only "Active" data is chosen, code pertaining to the "Archived" data need not be executed. To create conditional macro code, use the *%if…%then…%end* statement as seen in this example:

```
%if &files = Archived %then %do;
   data ARCH16;
     infile ARC16;
     input @1 zip … @211 uic $6.;
     call symput('arch16','ARCH16');
     ……….
   run;
 %end;
```

If "Archived" data is chosen, then the macro variable *&arch16* is assigned the value *"ARCH16"*, which is the name of this data set. Otherwise, it remains blank as assigned in an earlier DATA step. Thus, when *&arch16* is evaluated in later SET statements, the data set *ARCH16* will only be included in the DATA step if "Archived" data has been chosen.

SAS/CONNECT

To transfer report data between the mainframe or a Windows NT server, and the user's PC, connections between the systems involved in the transfer must be made via SAS/CONNECT. When the Standard or Custom Reporting Menus are entered no further processing is performed if the connection to the proper platform is not successfully established. The next example establishes a connection to the Windows NT server.

```
submit continue;
  options remote=nts1node comamid=tcp;
  filename rlink
   'c:\sas\connect\saslink\tcpwin.scr';
  signon nts1node;
endsubmit;
rc=rlink('nts1node');
```

The REMOTE= value, *nts1node*, was established in the *autoexec.sas* file for the reporting application as a valid SAS system name representing the address of the Windows NT server. The COMAMID= value identifies TCP as the communication method being used. The *rlink* filename identifies the logon script being used by SAS/CONNECT. After logging on to the Windows NT server, the RLINK function in SCL is used to check for a valid connection to *nts1node*, storing the function's return code in *rc* for later use.

The scripts used for logging on to different platforms via SAS/CONNECT are included with the SAS software. Many may be used "as is", but some changes had to be made to the *tcptso.scr* script to conform to our MVS mainframe connection requirements. Examples include changing the logon messages and the SAS startup procedure name to match those used by our mainframe.

STANDARD REPORTING MENU

The purpose of the Standard Reporting Menu is to provide the Army Hearing Conservation Program managers and consultants with ready access to reports used on a regular basis. Data for a user-specified report are downloaded to the requesting PC from the Windows NT server using PROC DOWNLOAD. After the data is successfully downloaded, the data is sorted by year and the available years are presented in a list box for selection. The data is then restricted to the user-specified year and the "Report Options" list boxes are populated for that report. Once all required variables are selected, the report is generated by clicking on the *"Run"* button. See *Figure A* for an illustration of the Standard Reporting Menu.

Figure A

CUSTOM REPORTING MENU

The purpose of the Custom Reporting Menu is to provide the Army Hearing Conservation Program managers and consultants with the ability to generate customized reports. After selecting the type of report desired, you must enter the type of data to be used as input, a date range, and a report option. See *Figure B* for an illustration of the Custom Reporting Menu.

Figure B

Following the selection of all desired options, report data are queued by clicking the *"Queue"* button, one report at a time, until all requested reports are entered. These reports are then submitted to the mainframe by clicking the *"Submit"* button. Completed reports may be viewed by clicking the *"Report Generation"* button and selecting a completed report.

Security

Since different users may submit numerous reports to be run on the mainframe in batch mode, a method was required to restrict an individual from viewing or deleting reports that were generated by others. This is accomplished by using the mainframe user id. Immediately after verifying a successful connection, the user id is retrieved from the mainframe using the automatic macro variable *&sysuid*, and then downloaded to the PC. If no errors have occurred in the procedure, the user id is

stored in the variable *masterid*. *Masterid* is then saved along with all report parameters for each report submitted. When accessing a report in any way, the current value in *masterid* is compared to the user id stored with the report parameters. Only those reports with a matching user id are accessible.

Report Submission

To pass the report selection options to the report macros on the mainframe, the values of the option variables are first stored in a report parameter library on the PC. The member names in this library correspond to the unique names assigned to the report when it is queued. This unique report name is created by concatenating the three-character abbreviation for the current weekday with the current time. Upon submission, each new member of this library is uploaded to the report parameter library stored on the mainframe.

Also stored on the mainframe is a JCL code submission library. This library contains JCL code that calls the report macros. Every custom report available in the reporting system has a corresponding member in this library. Each member contains JCL code that allocates the input data files, and base SAS system code that retrieves the report parameters and calls the report macro. To retrieve the parameters for a particular report, the CONTENTS procedure is run against the report parameter library to compile a list of the current members, storing the information in a data set called *content*. Then a DATA step loops through *content*, and for each member CALL EXECUTE statements create null DATA steps that look for a type of report identical to the calling report. Once found, the appropriate report parameters are saved as macro variables and are included in the call to the report macro.

To submit one of these batch reports from the Custom Reporting Menu, a LIBNAME statement is issued to assign the libref *injcl* to the JCL submission library on the mainframe. The following code is then remote submitted for each custom report using SAS/CONNECT.

```
filename injcl 'bad.hears.jcl.submit';
  %sysrput rmtret2 = &sysfilrc;
  %macro submitit;
    %if &sysfilrc = 0 %then %do;
       filename outrdr sysout=x
         pgm=intrdr recfm=fb lrecl=80;
       data _null_;
         infile injcl(&rpt);
         file outrdr noprint notitles;
           input;
           put _infile_;
       run;
       filename outrdr clear;
    %end;
  %mend;
  %submitit;
filename injcl clear;
```

The return code from the LIBNAME statement is contained in the automatic macro variable *&sysfilrc*. If the LIBNAME statement results in an error, the macro that submits the JCL code for the report will not execute. The *outrdr* file is the fileref for a special file used by the mainframe to submit batch programs. See *SAS Companion for the MVS Environment, Version 6, First Edition, page 227*, for specific details.

Status Reporting

Since the Custom Reporting Menu automatically logs you onto the mainframe, there is no way to directly monitor the progress of submitted reports running in batch mode. To provide this capability, the Custom Reporting Menu program and the macro report programs create messages regarding the progress of report generation and save them to a status data set stored on the mainframe. Each message contains the unique report name, a status field to indicate the type of message, the user id of the submitting user, and the current date and time the message is written. At the time of submission, a "Submitted" message is saved to a temporary data set on the PC and then uploaded to the mainframe status data set for storage. At run time on the mainframe, the JCL submission code writes a "Running" message to the status data set. Furthermore, if an error is detected after any major procedure or DATA step throughout the report macro, the %message macro writes an "Error" message to the status data set containing the code number of the error which has occurred. Finally, the last task performed by the report macro is to post a "Completed" message to the status data set, which also contains the number of observations in the completed report data set.

The status of the report jobs is monitored by clicking on the *"Status"* button on the Custom Reporting Menu. The status and report parameter data sets are downloaded from the mainframe with PROC DOWNLOAD and merged together by the unique report name. PROC REPORT is then used to generate a report that displays the related status messages and report parameters for all reports created by a user.

REPORT GENERATION

From the Standard Reporting Menu, selected report data are retrieved from a Windows NT server using a "where rpttype = &rpt" clause in conjunction with PROC DOWNLOAD, where *&rpt* is the name of the selected report. Further WHERE clauses to be used in the PROC REPORT are created using any report selection options specified by the user (e.g., zip code).

From the Custom Reporting Menu, a completed report is generated by clicking on the *"Report Generation"* button. Another menu then displays a DATA TABLE containing all reports submitted by that user. A successfully completed report is selected, and the corresponding data are downloaded from the mainframe with a PROC DOWNLOAD. Similar WHERE clauses are also created based on the corresponding report parameter data downloaded from the mainframe.

The same PROC REPORT is used to generate both standard and custom reports. The following is the basic framework for a PROC REPORT used in the Army HEARS Corporate Data Reporting System. Notice that the WHERE clauses (e.g., *&wcl1*), generated in the calling menu programs appear as macro variables in the procedure code. The BY statement in the PROC REPORT results in the report being displayed in order of the BY variables, with the values of these variables being displayed before the column headers on each page of the report. Other variables included in the report as analysis variables are referenced by their statistical function (e.g., *f1sts.sum*) in the COMPUTE sections.

```
proc report data=work.rptdata LS=175
            PS=67  split="*" headline
            nocenter missing nowd;
  &wcl1 &wcl2 &wcl3 &wcl4 &wcl5;
  by service site;
  column rpttype …… pnof2;
  define rpttype / order format= $8.
         width=8   spacing=2
         noprint left "Report Type";
  ……
  define pnof2/computed format=6.2
         width=9 spacing=1 center
         "% no   *test   *on F/u2 ";
  compute pnof2;
    if f1sts.sum=0 then pnof2=0; else
      pnof2=nof2sts.sum/f1sts.sum)*100;
  endcomp;
  break after site/dol page summarize ;
  break after service / page ;
  title1 'xxxxxxxxxxxxxxxxxxxxxxx';
  footnote1 'xxxxxxxxxxxxxxxxxxxx';
run;
```

To pass necessary parameters to the PROC REPORT, use an ENTRY statement before the INIT section of the calling SCL program. These variables can then be referenced within the SCL code, but they must be prefaced with an ampersand when referenced within code remotely submitted to base SAS software. When calling the report program, be sure to list the report parameters in the same order as in the ENTRY statement.

DATA ANALYSIS AND MANIPULATION

To store the report data for further analysis, the Army HEARS Corporate Data Reporting System allows the saving of data to a permanent file. The reporting utilities of SAS/ASSIST are an excellent way to further explore the report data, and may be accessed from either the toolbar or the main menu of the HEARS Reporting System. Since many of the managers and consultants were familiar with Microsoft Excel®, they also requested the ability to easily transport data to Microsoft Excel for further analysis.

Exporting Data to Excel

We first tried to export SAS data sets to Microsoft Excel using DDE statements, but were unsuccessful. We could not find documentation on the Microsoft Excel commands used in DDE statements issued from within base SAS software. This resulted in code that worked so intermittently that it was of no practicable use to the HEARS Reporting System. Furthermore, Microsoft Excel limits the number of imported records to 16,384 (i.e., the size of one worksheet). This was unsatisfactory for the HEARS Reporting System since some of the data sets may contain 80,000 or more records. Consequently, we abandoned the use of DDE statements and searched for a better solution. Fortunately, we found a solution on the Internet in the Knowledge Base at Microsoft's home page. Here we located a Microsoft Excel macro that imports text files over multiple worksheets and a macro that creates Excel toolbar buttons. Finally, we had a way for SAS data sets of any size to be imported to Microsoft Excel by calling the import macro via our custom Excel toolbar button. Although slightly complicated, it worked.

An improved and much simpler solution became available with the new Export facility incorporated in the release of the SAS System for PC's version 6.12. However, since Microsoft Excel still limits the number of imported records, we could not eliminate the Microsoft Excel macros altogether. So, we decided to combine the ease of use and reliability of the SAS Export facility with the large data set transporting capabilities of the Microsoft Excel macros to provide a complete data export solution.

The first step in the export process from the system's *"Send a Data Set to Excel"* menu is to select the data to be exported, at which time the number of observations in the data set is determined using the ATTRN function in SQL. The number of observations is then displayed on the menu using an INPUT FIELD object. Furthermore, regardless of the number of observations in the selected data set, you can always click the *"View Data Set"* button to browse the data in tabular form using the SAS System for PC's new VIEWTABLE utility.

If the number of observations is less than or equal to 16,384, then the *"Small Data Set Export"* button is enabled. Clicking this button opens the SAS System for PC's Export utility.

```
exptcmd = "dexport "||symget('dataid');
call execcmdi(exptcmd);
```

The DEXPORT command calls the Export wizard with the input data set identified by *dataid* in the preceding code. The wizard also prompts for the export format and the destination of the exported file. When the Export wizard is complete, a verification window for the execution of Microsoft Excel is presented. If you click "OK", then the SAS X command performs the execution. Once inside Microsoft Excel, the exported SAS data set may be opened with "File/Open", just as any other Microsoft Excel spreadsheet file.

On the other hand, if the number of observations is greater than 16,384, the *"Large Data Set Export"* button is enabled. Clicking this button converts the selected SAS data set into a tab delimited text file and opens Microsoft Excel. Once inside Microsoft Excel, you click on the "SAS" button we created which calls Microsoft Excel's import macro. The imported data is not parsed into columns, however, so you must do that yourself using Microsoft Excel's menus.

Intermediate Files

Another requirement for the Army HEARS Corporate Data Reporting System was to provide the Hearing Conservation Program managers and consultants with intermediate files in conjunction with certain reports. These files are generated during the creation of the report data, and contain specific information about individuals in the Hearing Conservation Program whose hearing test results match the report criteria. Both standard and custom reports may generate intermediate files, but they are generated in a custom report only if specifically requested via the Custom Reporting Menu. The generation of custom intermediate files is controlled by conditional statements in the report macros, where *&ival* is equal to 1 if intermediate files are requested.

```
%if &ival = 1 %then %do;
  %let iname=bad.hears.intmed.&rptdate;
  %str(libname INTERMED "&iname"
```

```
      disp=(new,catlg,delete) unit=user
      space=(trk,(2000,1000),rlse););
%end;
```

The intermediate file data sets are also conditionally declared and output in the DATA steps.

```
data TOTAWARD
 %if &ival = 1 %then
  INTERMED.OWCP(keep=zip …… compawd);;
 set TMPAWARD;
 ……
 output TOTAWARD;
 %if &ival=1 %then
   %str(output INTERMED.OWCP;);
run;
```

The intermediate files generated by standard reports are stored on a Windows NT server in libraries named with the year corresponding to the files' date range. To retrieve standard intermediate files from a Windows NT server, PROC DOWNLOAD is used with a SELECT statement which downloads only those members of the intermediate library corresponding to the selected report. These downloaded files are then stored in a user-specified library on the PC and remain until replaced or deleted.

The custom intermediate files are first stored in a library on the mainframe, as illustrated in the code above. The library name is distinguished by "BAD.HEARS.INTMED." concatenated with the unique name of the generating report, *&rptdate*. When a completed report is generated on the Custom Reporting Menu, any requested intermediate files will be downloaded along with the report data. The Custom Reporting Menu program creates a new library in which to store these files on the PC. The name given to this new library is based on the unique name of the report that generated the intermediate files.

```
intflout = "c:\sas\sashears\"||rptdate;
call symput('intlib','x mkdir'
            || intflout);
testit = symget('intlib');
call execcmdi(testit);
call display('sashears.custom.makedir',
              intflout);
```

In the above code, *rptdate* is the unique name of the report. The X command issues the MKDIR command to DOS in order to create the new library. The X command is first saved as a macro variable so that it may be created "on the fly." SYMGET is used to retrieve the X command from the macro variable, while the EXECCMDI command executes it. Unfortunately, DOS commands execute more slowly than SAS system commands. As a result, the SCL code continues its execution before the library is created, thus making references to a library that does not yet exist. Therefore, the CALL DISPLAY statement is a necessary part of this code. The *sashears.custom.makedir* program displays a window on the screen to notify you of the library creation, and requires you to click "OK" to continue. While the SAS System is waiting for your response, the DOS MKDIR command has ample time to execute. The custom intermediate files remain on the mainframe and the PC for twenty days, along with the corresponding report data, after which time they are all deleted.

BELLS AND WHISTLES

Here are just a few tips on how to add that "little something extra" to your application.

HEARS Reporting System Icon

To create an icon used to call the Army HEARS Corporate Data Reporting System, we created a new shortcut in Windows 95, and included the following code in the properties of the icon.

```
C:\SAS\sas.exe -autoexec …… -config ……
```

The *-autoexec* and *-config* options, followed by actual filenames, call the autoexec.sas and config.sas files created for the HEARS Reporting System. In addition to the regular config.sas code, the HEARS Reporting System config.sas file contains special instructions pertaining to its operating environment. For example, the *-awstitle* command establishes the title of the SAS display manager window as the title for the HEARS Reporting System, and the *-noawsmenumerge* command instructs the SAS System not to merge the SAS system menu bar with the HEARS Reporting System menu bar.

The autoexec.sas file contains code that establishes SAS options specific to the HEARS Reporting System (such as, format locations and fonts), and assigns libnames to critical libraries. In the following code, the DM statements maximize the windows, turn the log and program windows off, and call the HEARS Reporting System Main Menu.

```
options noxwait noxsync
        fmtsearch=(sashears.formats)
        font='Courier New' 10;
libname sashears 'c:\sas\sashears';
libname intermed 'c:\sas\intermed';
dm 'awsmaximize on'; dm 'log off;';
dm 'af c=sashears.hearsapp.main.frame;
    pgm off;' af;
```

Logo Screen

To replace the SAS logo screen with a logo of your own that appears when your application executes, simply add the following code to the beginning of the *config.sas* file.

```
-splashloc c:\sas\sashears\logo.bmp
```

System Menus and Toolbars

The PMENU procedure is used to create a custom menu bar. Your custom menu bar may be integrated with the SAS system menu bar or used alone, depending on the options you add to the *config.sas* file. For more details, reference the *SAS Procedures Guide, Version 6, Third Edition*.

Use the TOOLEDIT command on the command line to invoke a window that allows you to create a custom toolbar. In the tool edit window you can add, manipulate, and delete buttons for the toolbar. Most toolbar buttons execute SAS system commands, but if you need to call another SAS/AF program from within an existing SAS/AF session you must use the AFA command instead of the AF command. For further details please see the

SAS Companion for the Microsoft Windows Environment, Version 6, First Edition.

CBT Help Entries

To create an efficient help system for the Army HEARS Corporate Data Reporting System, we created CBT entries using the SAS text editor. CBT entries may be divided into many different frames that appear on your screen one at a time. A LOCK statement, followed by the name of the frame, is used to delimit the CBT entry's frames.

To change the color attributes of the text in a CBT entry, use the COLOR TEXT command on the command line. This command should be followed by a color name and any color attributes you want to assign to the text, such as "COLOR TEXT GREEN UNDERLINE."

You can identify specific portions of text as hypertext links to other frames within the CBT entry. The portions of text you want to enable are numbered in the order they appear on the screen. These numbers are followed by characters indicating how the path through the frames is to be remembered when backtracking, and the name of the CBT entry and frame that is to be displayed. The final SELECT= option denotes the beginning and ending column and row numbers for the section of text to be highlighted when selected. For more information on CBT entries, please refer to *SAS/AF Software Usage and Reference, Version 6, First Edition.*

To call a CBT entry as a help screen for an object in a FRAME entry, you must specify the name of the CBT entry in the Attribute Window for that object. You may also specify the name of a frame within the CBT entry that will display when help is requested for that object. However, if you happen to name one of your frames the same as an item within the SAS system help, that SAS system help window will appear instead of your own.

CONCLUSION

The development of the Army HEARS Corporate Data Reporting System application evolved over a two-year period. Prior to and throughout this time, we attended many of the SAS User Group conferences to obtain tips, techniques, and insights from other SAS users who may have developed similar applications or have used the many SAS tools we sought to use in a client/server environment. For example, we obtained some valuable tips from a Solutions@Work™ CD-ROM distributed at the NESUG '96 conference that helped a great deal in engineering the DATA TABLE objects now used in the HEARS Reporting System. We learned many things from our peers and, after many trials and errors which lead to numerous calls to SAS Technical support, we learned even more from our own experiences. Now that the majority of the Army HEARS Corporate Data Reporting System is complete, we decided to compile and share many of the tips and techniques we found to be most beneficial in hopes of saving others a few hours of extensive research and frustration.

REFERENCES

SAS Institute Inc. (1989), *SAS/AF Software, Usage and Reference, Version 6, First Edition*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1993), *SAS Companion for the Microsoft Windows Environment, Version 6, First Edition*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1990), *SAS Companion for the MVS Environment, Version 6, First Edition*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1990), *SAS Procedures Guide, Version 6, Third Edition*, Cary, NC: SAS Institute Inc.

"XL: Importing Text Files Larger Than 16384 Rows." *Microsoft Technical Support Knowledge Base.*
15 OCT 1996.
http://www.microsoft.com/kb/articles/q120/5/96.htm (2 DEC 1996).

"XL: Setting Status Bar Text and ToolTips for Toolbar Buttons." *Microsoft Technical Support Knowledge Base*.
13 SEP 1996.
http://www.microsoft.com/kb/articles/Q112/6/32.htm
(3 DEC 1996).

For more information about this paper contact:

Krista Elspas at (301) 619-7703 or email:
Kris_Elspas@ftdetrck-ccmail.army.mil