Managing Information Access and Updates

Daryl Hoffman
Resource Administration Group
Center for Academic Computing
Penn State University

Overview

This paper describes how SAS® can be used to manage information for an academic accounting system. SCL and new access engines have made it easier to maintain an information system that has become much more diverse and less platform dependent. SAS® products enable the creation of an information system with easy access, maintenance and data security. The paper will cover the interactive applications used to provide these features, briefly touching on the accounting system used at the Center for Academic Computing (CAC), which currently supports over 20,000 clients. The data is used by the Center to provide University administrators with information pertaining to the use of computer resources.

Although this paper deals with one specific SAS® system at Penn State much of the information can be used in other academic and professional institutions and tailored to meet the needs of the specific user base.

Introduction

The accounting system involves two main entities, the account portion and the user information portion. The account data set contains the charge-back information such as balance, authorization, supervisor name and address. The user data set contains the userID, initial password, and disk size. The two data sets are tied together by the account number. The main function of the accounting programs is to track the usage of the mainframe for printing, batch job submission and printing. The accounting system is a charge-back system in which the charges are applied to the account daily. The accounts are opened and maintained with the use of SAS/AF and SAS/FSP. The user information database keeps all of the user profile data: name, social security number, address, phone, and a unique field known as userid. Each of these two main collections of data are maintained with the use of interactive applications developed with SAS®, SAS/AF®, SAS/FSP® and SAS/ACCESS®.

Evolution of the system

Before implementing the accounting system with the SAS System for Information Delivery, the accounting system consisted of programs maintained in PL/I, FORTRAN G, REXX and REXX Display Manager, Assembler, and even one program written in COBOL. As the accounting system grew from several hundred mainframe accounts into several thousand accounts with multiple users on each account, the need to find a common system for collection, reporting and maintenance became a top priority at the Center for Academic Computing. After twenty years of production the existing accounting system was out of date and the system had to be made more flexible and easier to maintain.

After investigating many vendor packages that did what was necessary for the accounting system and much more, it was suggested that the vendor applications had several things in common. Most were using COBOL or SAS® to do the data reporting and manipulation for the accounting system. Also, many were using Barry Merrill's code, MXG Software, for collecting the SMF and other machine usage data and creating usable SAS® data sets. The Resource Administration Group then decided to use MXG® as the base for the accounting charge collection and then use SAS® to generate reports, update data and other tasks. The first system was written with SAS® 5.18 by several programmers who knew very little about the capabilities of SAS® could do or how to use SAS® to produce the desired information. The data was to be stored in SAS® data sets on CMS minidisks and the access to the data was to be controlled by SAS/SHARE®. The data entry would be handled with SAS/AF® and the SAS/FSP® products. The only problem was there was no easy way to check the data for correctness before the records were added to the data set. This was judged to be a noncritical item and was simply set aside for further investigation after the implementation of the system. After a few months of development and testing by the programmers it was time for testing by the clerical staff, the actual users of the

accounting information system. From the first minutes of the live test run, problems were encountered with record locking and concurrent access. This was a major setback because the clerical staff had to be able to update and access the data concurrently during their daily routine of updating and adding new accounts. There seemed to be no viable alternative until SAS Institute announced the release of version 6.06 of the SAS® system for CMS and other platforms.

Converting from SAS® 5.18 to SAS® 6.06

After the new version of SAS® was installed on CMS, the programming staff went to work. The only major program conversion involved the SAS/AF® applications. The applications catalog was converted using PROC V5TOV6. The conversion had only a few problems that were overcome with the help of SAS Institute's technical support staff. The next step in the conversion was getting up to speed on SCL, the language used to control the flow of the applications as well as adding the capabilities of error checking and data validation.

Production

With the installation of SAS® 6.06 it was decided that the data would be stored in SQL/DS™ tables and the access to the data would be achieved with a new product called SAS/ ACCESS®. After creating the views for the SQL/DS™ data, and making changes to the SAS/AF® applications, we were ready for another production test run. With the implementation of SCL in Version 6 of the SAS® system, the problems of providing error checking, data access from the previous release of the SAS® system were overcome. The accounting system was now in production and all of the applications were written with SAS/AF®, SCL and SAS/FSP®.

The batch processing was written with the use of user written macros and Base SAS® code. The charges were written to CMS minidisks for access and report generation in SAS data set format. The accounting data sets were downloaded into the SQL/DS™ tables in the early morning so that there was no problem with users of the system being 'locked out' because of SQL/DS™ table locking.

Menus

The program was created to provide an easy to use interface for the clerical staff. To guide the clerical staff through the different application programs, BLOCK menus were created. Initially they did not contain the graphical icons that are now being used by the SAS® 6.07 and later releases of SAS® software and they were set up as *program* entry types. With the release of SAS® 6.07 the icons were added to the menus and several of the *program* entries were converted to *SCL* entry types. The only difference in the actual SCL code was the AF entry type. The SCL entry type is the same as a PROGRAM entry type with the exception of the display screen. SCL entries do not allow you to provide the user with a screen contaning SCL or SAS data set variables. SCL entries are used mainly for menus and dialog boxes that do not require a screen that accepts input of information from the user.

In addition to the BLOCK menus, the program also took full advantage of choice groups and dialog boxes. The choice groups were used to allow the user to choose from a list of options and then execute a specific application program or other SCL functon that was not suitable for creating a MENU or BLOCK entry in the AF application. To set up a choice groups you need to select the attributes window in your AF program and then choose a variable type of either ACTION or *PUSHBTNx*. The choices are tied together with the *choice group* part of the attribute window.

Example:

Field name: FIELD1 Frame: 1 Row: 5 Col: 14
Length: 1
Alias: FIELD1 ***Choice group:*** *ACCTTYPE Pad:*
Type: ACTION Protect: NO
Format: Just: LEFT
Informat:
Error color: RED attr: REVERSE Help:
List:
Initial:
Replace:
Options: AUTOSKIP

Field name: AACCT Frame: 1 Row: 5 Col: 16
Length: 11
Alias: AACCT ***Choice group:*** *ACCTTYPE Pad: _*
Type: CHAR Protect: YES
Format: Just: LEFT
Informat:
Error color: RED attr: REVERSE Help:
List:
Initial: A Account
Replace:
Options: AUTOSKIP

Field name: FIELD3 Frame: 1 Row: 6 Col: 14
Length: 1
Alias: FIELD3 Choice group: ACCTTYPE Pad:
Type: ACTION Protect: NO
Format: Just: LEFT
Informat:
Error color: CYAN attr: REVERSE Help:
List:
Initial:
Replace:
Options: AUTOSKIP
...

Pull Down Menus (PMENUs)

Enhancements have been added to the FSEDIT® screens which give more flexibility to the clerical staff. One of the enhancements added was the use of PMENUs. The menus are created using the Base procedure, PROC PMENU. The PMENUs consist of dialog boxes and valid SAS® commands that the users of the applications are permitted to issue and the SCL code is designed to process. Examples of the valid commands are END, CANCEL, HELP, and others. Other commands are also available that were specific to this application only. In order to process these commands you need to specify CONTROL ALWAYS in the FSEINIT section of your SCL code. The PMENU is also turned on in this section. PMENUs make it easy for the user of the application to navigate through the information system without having to remember and type a multitude of valid SAS® and host system commands. The commands are displayed for the user to easily see and execute.

Example:

```
FSEINIT:
  rc = pmenu('libref.catalog.menu_name.pmenu');
  call execcmd('command'); /* turn on pmenu */
  control always; /* allow for capture of all user
commands
                  */
RETURN;
INIT:
...
RETURN;

MAIN:

  If word(1) = 'user' then call display('af program');

RETURN;
```

PMENU code:

```
proc pmenu cat=libref.catalog;
  menu mymenu;

  item 'Help' ;
  item 'Finished' selection=done;
  item 'Commands' menu=cmdmenu;

  selection done 'end';

  menu cmdmenu;

  'Host commands' selection=hostmenu;
  'SAS command' dialog=commands;
  'SAS data libraries' selection=library;
  'SAS/ASSIST" selection=assist;

  selection library 'lib';

  dialog commands '@1';

   text #1 @4 'Enter valid SAS command:';
   text #1 @45 len=15;

  selection assist 'ASSIST';

  menu hostcmd;

...
```

The above code works under version 6 releases of SAS®. The code was tested under both 6.07 and 6.08 running under the VM/ESA operating system (CMS). The PMENUs are controlled under the GATTR portion of the AF program or they can also implement similar code to that previously listed by placing the activation of the pmenu in the INIT section of the SCL program. With version 6.08 of the SAS® system the PMENUs can also be made to work as a pop-up menu. This is something that is planned as a future part of the information system described in parts of this paper.

Data Entry Applications

The data entry applications used for adding new accounts and maintaining the old were driven by using SAS/AF®. The application made use of block menus, choice groups and data entry screens. The user of the acounting system did not have to know anything about SAS® or any of its components. SCL and the checking that was provided by

the SCL code helped to guarantee the integrity and correctness of the data that was entered or updated in the accounting system.

When the user entered a search parameter the values entered were verfied before the observation was displayed. Checks were also added to ensure that duplicate values were not encountered. The checks were done before all of the information was entered to avoid the need to rekey in the information when the insert into the SQL/DS table was denied because of a unique key conflict. The checks were accomplished with the use of the SCL functions LOCATEC, OPEN, and FETCH.

As the users of the accounting system became more knowledgeable of what the programmers could provide with SAS® and the applications development tools in SAS/AF® and SCL, more features were added and existing functionality was enhanced to take full advantage of the power of the SAS® system.

Maintaining the Database

The data was maintained using FSEDIT®, a part of the SAS/FSP® product. The applications progammer developed the screens and provided the user with a pre-defined data entry screen that was to be completed before the data was appended to the SQL/DS table. SCL allowed the programs to provide address validation, field validation and other data integrity features. The user was also presented with pull-down menus that were created using PROC PMENU a procedure which is part of Base SAS®. The menus contained search capabilities, the ability to display information about the users of the accounts, and also a subset of commonly used host commands.

Other Application Functions

In addition to maintaining up to date accounting information the clerical and programming staff also had the ability to display charges, transfer charges, print the forms necessary for maintaining user accounts, and many other specific functions.
The SCL functions FETCH, FETCHOBS, LOCATEC, DATALISTC, PUTVARx and GETVARx were heavily used to retrieve and update the information. The flow of information from the many different data sources was all controlled from within the application itself. In release 6.07 of the SAS® system the list functions were enhanced with the addition of the MAKELIST, CURLIST and GETNITEMx. These functions were added to improve several of the existing functions such as DATALISTC and SHOWLIST. With the new list functions the applications programmer had more control over the way the SCL code was executed and the parsing of the choices made from the list functions was made easier.

Example:

```
MAIN:
...

listid = makelist();
rc = curlist(listid);

call wregion(7,7,15,40,'');

dataset = datalistc('library','data',9,'Y','','','Select the data
set
          to view');

n = getnitemn(listid,'COUNT');

do i = 1 to n;
   member = getnitemc(listid,'NAME',i);
   call fsview(member,'browse',formula);
end;
...
```

Audit Trails

With the possibility of up to six people updating the data at any time throughout the day, it was necessary to provide some form of tracking the updates made to the data. The audit trail kept track of what fields were changed, when they were changed (date and time), and the previous values for the modified fields. The audit record also contained the userID of the person responsible for the update. The audit trail was created not so much to know who to blame as it was to know what was changed on the particular observation. This information was previously maintained on microfiche on a monthly basis; any changes occurring on a daily basis were lost. The audit trail also provides a way to recover any lost or missing records after an accidental deletion.

Problems Encountered

The problems that were encountered dealt mostly with concurrent access to the data. The two main data sets were stored in SQL/DS. The information stored in SQL/DS was accessed using access views created by using the SAS/ACCESS® product interface to SQL/DS. The main

problems with this way of storing the data was that you could not simply add, updated and delete records and have the indexes remain intact. A program was written that executes in the overnight processing in batch mode which extracts the current data from SQL/DS and applies all of the updates. After all of the programs have applied the updates to the extracted data set, the dataset is reloaded into the SQL/DS table. The table is recreated and reloaded daily. The audit trails were stored on minidisks. This would have presented a problem since we did not license SAS/SHARE®. In order to avoid multiple write access problems, the shareable data was stored on the CMS Shared File System (SFS). With SFS, multiple users can update the contents of the same file space at the same time. SFS also provided security through the use of GRANT commands that are part of the SFS and the CMS operating system.

The other problems were mostly in training the users of the application. The clerical staff was not used to being able to update the information with the touch of a button and was more concerned with mistakes. After gaining the experience of using the system and seeing the power that was now theirs, they began to adjust to the new information system.

The Future

With the current release of SAS®, version 6.08, the potential seems unlimited for maintaining the accounting information at the CAC. One possibility now under consideration is that of unloading the report generation off onto the Windows or OS/2 platform. With the use of SAS/CONNECT® the data can be easily downloaded to the Windows environment and the reports can then be generated and printed on laser quality printers as well as color graphics devices.

Also being investigated is the new FRAME entries of SAS/AF®. This new entry type will allow the programmer to create a Graphical User Interface (GUI) for the users of the accounting system.

In the future we hope to be able to use SAS® to develop an EIS that can be used by the entire Center. On-line application forms and direct billing used to be planned for the distant future, but this can now become reality and will be
easier to implement across platforms. With the exit of the mainframe as we know it today coming within the next decade, plans are in the making for transporting and accessing data across different platforms in a LAN in the office or across the University. Client/server technology has become a new buzzword, and students are asking for access to the newer technology. In order to meet the overwhelming demand for accounts for every student and prevent the need for more personnel, on-line applications will become a major part in the assignment of new accounts and userIDs. With the client/server software SAS/CONNECT®, the on-line applications could be placed in any of the computer labs that have access to the data stored on the mainframe. This is accomplished by exporting the AF catalogs to the platform that is to be used in the computer labs, and changing the AF applications that access the mainframe data toouse RSUBMIT blocks and verify the data entry at the workstation. This eliminates the need for human intervention. In the future, a user could receive the new account, userID and password at the workstation without the need to wait for clerical staff to process the request.

Now that all of the Center accounting is done with SAS, the applications and program maintenance are easier. Why ? The learning curve is almost non-existent. The only programming language that one needs to be knowledgeable of is SAS®. The programs are driven by AUTOEXECs and REXX on the CMS platform. Similar means of invoking the application are anticipated for the other platforms that will be used in the future.

SAS Institute is very committed to platform independence, and this is very important in the University environment because of the ever changing user platforms. With the development of vendor architecture shifting away from the mainframe time-sharing systems to the individual workstations, the accounting system at the Center for Academic Computing at Penn State can be easily ported to the current platform with the use of SAS®. All of the data sets and program catalogs can be exported to the chosen platform and then imported with ease. The SAS® system for Information Delivery was a great choice that will benefit the CAC for the life of the existing accounting system and beyond.

References

SAS/AF® Software: Usage and Reference, Version 6, First Edition

SAS/FSP® Software: Usage and Reference, Version 6, First Edition

SAS® Screen Control Language: Reference, Version 6, First Edition

SAS® Screen Control Language: Usage, Version 6, First Edition

SAS® Technical Report P-216, Changes and Enhancements to SAS/AF®, SAS/FSP and SAS® Screen Control Language

SAS® Observations, Fourth Quarter 1991, "Journaling an FSEDIT Session"

SAS®/CONNECT Usage and Reference

SAS® Technical Report P-224, SAS®/CONNECT Software: Changes and Enhancements, Release 6.08

Advanced SCL Applications Course Notes (Release 6.07)

SAS/ACCESS® Interface to SQL/DS, Usage and Reference, Version 6, First Edition