A BLUEPRINT FOR INTERACTIVE PC-BASED APPLICATIONS DEVELOPMENT

Alman M. Zeld
HAY Systems, Incorporated

INTRODUCTION

Developing a software application system is probably the most challenging and rewarding task in the Data Processing career. Software developers have a long list of issues and problems to consider and resolve. The contents of this list, however, vary depending upon the size of the project, the type of the application, and the language used, among other factors. The intention of this paper is not to provide this long list of issues, nor to recommend solutions for those problems. The intention, however, is to discuss and highlight some common topics specific to any interactive PC-based software application development effort using the SAS® system version 6.04.

This tutorial includes the following topics:

1. General and essential steps in the PC application development process
2. Hardware and Software requirements
3. Overview of the SAS system files and subdirectories
4. File structure and naming conventions for your application
5. Overview of essential SAS statements and SCL functions used in any application
6. Where does the output go and what can we do about it
7. Important DOS files (CONFIG.SAS and AUTOEXEC.SAS) and how can we use them advantageously
8. Integrating all previous pieces to work as one system

1. GENERAL AND ESSENTIAL STEPS IN THE PC APPLICATION DEVELOPMENT PROCESS

Every software development team approaches the application development process in a slightly different way. However, all approaches and techniques share primarily the same basic steps. Depending upon various factors, these steps take different shapes and priorities. Some of these factors are:

- The depth of the client's understanding of the systems requirements and purposes. Does the client know exactly what he/she wants to develop?
- The extent of the client's technical knowledge determines the degree of the client's influence and participation in the development process.
- The project duration and resources play a significant role in selecting the development approach.

Generally, the following steps will be included in any approach to the process of systems development:

DEFINING THE SYSTEM REQUIREMENTS

What exactly are we trying to accomplish? Is it one or two types of analyses? Or is it a list of ten or twenty reports? Maybe it is a combination of the two. It also could be an on-line information system that is used to store, update, and merely display information on the screen. These are the types of questions we, as systems developers, should be asking ourselves and our clients. It is very significant, however, to insure that we, the developers, and the clients have the same exact answers to these questions. Experience has shown that it is a good practice to document these answers in writing and use them as a reference throughout the development process to keep directing the work toward accomplishing the agreed upon goals.

IDENTIFYING ALTERNATIVE SYSTEMS DESIGNS

The most efficient system design is probably reached by allowing the members of the designing team time to study the systems requirements and purpose. Each member of the team will then produce a design approach. At a brain-storming session, all members present their design ideas and approaches. The entire team participates in the process of validating the design ideas, studying the pros and cons for every approach, and finally selecting the most promising one.

DEVELOPING A DETAILED SYSTEM DESIGN

The selected design approach should be studied in terms of hardware and software requirements. The hardware analysis should include a selection of one item from each one of the following groups:
Using the PC environment, compared to the Mainframe software with COBOLs that are much lower than the Mainframe environment. Generally in terms and SAS/FSP product, we should begin.

Adjusting the design frequently to review the design during the development process and constantly adjusts and refines the design based on problems, concerns, and/or observations that may surface during the process. Adjusting and refining the design is almost inevitable and should not be avoided. The outcome of the adjusting and refining process is always a better and more flexible design. As a result of this process, in some cases, the developers might resort to bigger, faster and more powerful hardware to accommodate the growing design. Such a change should definitely be discussed with the client.

**OBTAINING FEEDBACK FROM THE CLIENT**

Meeting with the client as frequently as possible eliminates facing problems at the end of the project when little or no time is left, and where changes are very costly, time consuming, frustrating, and damaging to the relationship between the client and the development team. During such meetings, the developers will be demonstrating completed or nearly completed modules of the system to the client. The developers would then seek feedback from the client to make sure that the development process is going in the right direction. It is very important to reach an agreement about the completed modules. Again, experience has shown that a good practice would be to document the outcomes of those meetings in writing. Those documents would be used as references by the developers during the remainder of the development process. Discussions during such meetings could also, in some cases, result in expanding the scope of the system being developed and change the project duration. Meeting with the client should continue throughout the development phase.

**DEVELOPMENT OF SYSTEMS DOCUMENTATION**

After developing and thorough testing, the documentation phase should start. Two types of documentations, a Systems documentation and a Users documentation, should be developed. The Systems documentation describes all the components of the system, file structure, data set names and contents, and every technical aspect of the system that will be required to maintain it. Since the Systems documentation will be used by technical staff, the use of technical language in writing the Systems documentation is appropriate. The Users documentation, on the other hand, will be used by the system users. This documentation describes how to operate and use the system. While writing this documentation, the developers must keep in mind that the readers of this documentation may not be computer oriented. A simple language style that does not use technical expressions unless necessary, should be used in writing the Users documentation.

**INSTALLATION AND TRAINING**

The last step of this process is installing the completed system. This step not only includes installation, but more importantly, testing the system using the client's hardware and software; which should be identical to what was used during the development. One of the problems that might surface is a variation in the way the SAS system is installed in the two environments. Finally, and most importantly, is training the client. The entire effort might not produce the expected results if the client is left with a system he can not operate. Training the client ensures that the correct and most efficient method of operation is used.

In addition to these steps, the developers must give consideration to the following issues:

- The PC environment has different types of limitations. Disk space, available memory, and CPU capability are some of these limitations. The developers must fully understand these limitations and perceive every aspect of the system design with these limitations in mind.

---

**PC Microprocessor:**

- 286 Personal Computer
- 386 Personal Computer

**PC Memory:**

- 640 standard memory
- Expanded memory (How much?)

**Hard Disk Required Space:**

- 20 MB (How many drive partitions?)
- 40 MB (How many drive partitions?)
- 60 MB (How many drive partitions?)

**Graphics Cards:**

- VGA graphics card
- EGA graphics card
- CGA graphics card

**Output Device:**

- Impact Printer (What kind?)
- Laser Printer (What kind?)
- Plotter (What kind?)

The design, hardware and software requirements should be discussed with the client to obtain an approval. Using the PC environment, compared to the Mainframe environment, offers more options in terms of hardware and software with costs that are much lower than the Mainframe environment. Generally, in terms of the SAS system products, the SAS base product, SAS/AF product, and SAS/FSP product would be used in the majority of the interactive PC applications. At this point a full scale design should be outlined and the development process should begin.

**REFINING THE SYSTEM DESIGN**

It is very important that the development team meets frequently to review the design during the development process and constantly adjusts and refines the design. During such meetings, the developers will be demonstrating completed or nearly completed modules of the system to the client. The developers would then seek feedback from the client to make sure that the development process is going in the right direction. It is very important to reach an agreement about the completed modules. Again, experience has shown that a good practice would be to document the outcomes of those meetings in writing. Those documents would be used as references by the developers during the remainder of the development process. Discussions during such meetings could also, in some cases, result in expanding the scope of the system being developed and change the project duration. Meeting with the client should continue throughout the development phase.

**DEVELOPMENT OF SYSTEMS DOCUMENTATION**

After developing and thorough testing, the documentation phase should start. Two types of documentations, a Systems documentation and a Users documentation, should be developed. The Systems documentation describes all the components of the system, file structure, data set names and contents, and every technical aspect of the system that will be required to maintain it. Since the Systems documentation will be used by technical staff, the use of technical language in writing the Systems documentation is appropriate. The Users documentation, on the other hand, will be used by the system users. This documentation describes how to operate and use the system. While writing this documentation, the developers must keep in mind that the readers of this documentation may not be computer oriented. A simple language style that does not use technical expressions unless necessary, should be used in writing the Users documentation.

**INSTALLATION AND TRAINING**

The last step of this process is installing the completed system. This step not only includes installation, but more importantly, testing the system using the client's hardware and software; which should be identical to what was used during the development. One of the problems that might surface is a variation in the way the SAS system is installed in the two environments. Finally, and most importantly, is training the client. The entire effort might not produce the expected results if the client is left with a system he cannot operate. Training the client ensures that the correct and most efficient method of operation is used.

In addition to these steps, the developers must give consideration to the following issues:

- The PC environment has different types of limitations. Disk space, available memory, and CPU capability are some of these limitations. The developers must fully understand these limitations and perceive every aspect of the system design with these limitations in mind.
Because of these limitations, the PC environment may or may not be the correct choice for all applications. Selecting the appropriate environment is the most significant decision in the application development process.

Planning the system capabilities and design must be based on either sound and tested technical methods or strong previous experience by the developers.

Relying on any additional PC software in designing a PC SAS application system should be avoided. Any additional PC software may or may not interact properly with the SAS system unless extensive testing has been performed to suggest otherwise. Also, since most PC software are constantly updated by their vendors, future testing and possibly modifying, will be required every time a new release of the PC software or the SAS system is to be implemented.

This was an attempt to highlight only the basic and general steps of the process of PC-based application development. A process that always proves to be dynamic, challenging, and very rewarding.

The remainder of this tutorial will highlight and discuss topics that are specific to application development using the SAS system version 6.04 in the PC environment. Software application developers, who will operate in this environment, will have to become very familiar with these topics since they directly impact the development effort.

2. HARDWARE AND SOFTWARE REQUIREMENTS

One fact about the SAS system we are very familiar with, is its excellent features such as versatility, flexibility, and portability. Another fact we are also very familiar with, is that the amount of memory required to run SAS applications vary depending on the complexity of the application. For those of us who are using a mainframe system, the memory requirement is not a major concern. On the other hand, those of us who are using the SAS system in a PC environment, have to pay very close attention to this topic. The following factors have a direct impact on the memory requirement to run a SAS application in a PC environment. These factors also affect the speed at which the SAS application runs in a PC environment.

- The number of SAS data sets and SAS catalogs that must be open at any given time during the operation of the application
- Utilizing SAS file caching and SAS input and output buffers
- The amount of available memory above the conventional 640K (for instance 1 MG vs. 2 MG)
- The kind of personal computer used (for instance a 286 PC or a 386 PC)
- The SAS system software used in the application, and the combination of software which are active at any given time
- The amount of available hard disk space in the hard disk drive where the application resides
- The size of SAS data sets and catalogs
- The SAS procedures used in the application (for instance, PROC SUMMARY and PROC TABULATE)
- Data manipulation complexity that takes place during the application execution
- Other memory resident software that are installed in the personal computer and active while the SAS application is running

All these factors have a very noticeable impact on the performance of the SAS application. The higher the number of SAS data sets and catalogs that are opened by the application, the slower the performance. This is something that should be considered by the software developers while designing the file structure and the SAS catalogs used by the application. An efficient design, which minimizes the number of data sets and catalogs, will result in a much better performance.

Utilizing file caching, input and output buffers improves the performance of the application when used properly. Utilizing any additional memory above 640K also improves the performance of the application. As a general rule, adding more memory is strongly recommended when the SAS/AF and SAS/FSP software are used in the application. A discussion of how to configure the SAS system so it will utilize the extra memory you might have is presented in a later topic.

It is obvious from the previous discussion that we need the fastest and most powerful PC. Recommending a 386 PC over a 286 PC is a conclusion that most of us have already arrived at. It provides a faster disk access and faster microprocessor which drastically improves the performance of the application.

The more SAS software used by the application at any given time, the slower the performance. This is because the SAS system requires portions of the memory to load and run every software in addition to the SAS base software. For instance, in a SAS/AF application, opening
a SAS FSEDIT screen on top of an AF screen will require more memory in addition to decreasing the performance.

The SAS system uses the hard disk space as a work area for temporary data sets and other system files and catalogs. The more disk space available to the application, the faster it will run. The amount of disk space required to install the base SAS software and some additional products is:

- Base SAS software without EMS image: 5.28 MG
- Base SAS software with EMS Image: 5.78 MG
- SAS/AF software: 0.68 MG
- SAS/FSP software: 0.80 MG
- SAS/STAT software: 4.09 MG
- SAS/GRAPH software without maps: 4.31 MG
- SAS/GRAPH software with maps: 7.20 MG
- SAS/IML software: 0.69 MG

Using large SAS data sets and catalogs decreases the performance of the application. Complicated data manipulation requires more memory resources and hard disk space. Finally, running other PC memory resident software, while the application is operating, reduces the amount of memory available to the SAS system and consequently decreases performance.

The SAS system software required to develop an application varies depending on the purpose and type of the application. The SAS base software is, of course, essential.

Using SAS/AF software is the most powerful and versatile approach to develop menus and screens. In version 6.04 of the SAS system, the developers have to use a new language (Screen Control Language) which basically controls and operates the AF screens. The Screen Control Language adds a tremendous amount of flexibility and capability to the SAS/AF software. With the combination of SAS/AF and SCL, developers can open more than one screen at any given time, perform additional read and write operations on SAS data sets, open FSEDIT screens from within AF, and many other powerful features.

The use of SAS/FSP software is the only method of providing the user with interactive access to SAS data sets. This software provides editing and browsing capabilities, in addition to extensive screen variable verifications features. The Screen Control Language can be used behind the FSP screens to provide additional capabilities.

Other SAS software may be used as needed. For instance, SAS/OR can be used if there is a need to perform Operations Research Analysis or to produce a project network diagram using PROC NETDRAW. The SAS/STAT software may be used if there is a need for advanced statistical procedures.

3. OVERVIEW OF THE SAS SYSTEM FILES AND SUBDIRECTORIES

Since the PC environment is relatively small, it is important that we become familiar with the SAS system file and subdirectory structure so we will not accidentally override or erase any of the important SAS files. Installing the SAS system can be done using one of two methods. During either of the two methods, fast install method or custom install method, the installation process creates several subdirectories and stores several files in those subdirectories. Both methods of installation ask the user to specify the name of the subdirectory where the SAS system should be installed. For instance, if the user specified that the name of the subdirectory should be SAS and that drive C should be where the SAS system resides, the installation process will take control from that point on and create the following subdirectories under the C:\SAS subdirectory:

- C:\SAS\SASEXE : Contains executable SAS system files
- C:\SAS\SASHHELP : Contains SAS help files
- C:\SAS\SAINST : Contains files used in the installation process
- C:\SAS\SASLINK : Contains SAS micro-to-host link files
- C:\SAS\SASMSG : Contains SAS message files
- C:\SAS\SASUSER : Contains SAS user profile catalog
- C:\SAS\SASWORK : Used to store temporary SAS files during the execution of the SAS system

All these subdirectories are important to the SAS system because they contain essential files needed to run the software. The last two subdirectories, however, are specific to each SAS session. The SASUSER subdirectory contains the user profile catalog which stores function key settings, window attribute settings,
and notepads. The user can change the default attribute settings and store the new settings in the user profile catalog. The SASWORK subdirectory is used as a work space to store temporary SAS files during each SAS session. The two subdirectories do not contain any executable SAS files. Each time the PC user invokes SAS, the SAS system takes one of the following actions by default:

- If there are no SASUSER or SASWORK subdirectories in the current active DOS subdirectory where SAS was invoked, the SAS system creates two new subdirectories under the current active DOS subdirectory and calls them SASUSER and SASWORK. A default user profile catalog will be stored in SASUSER.

- If the two subdirectories, SASUSER and SASWORK, already exist in the current active DOS subdirectory where SAS was invoked, the SAS system will use them.

These actions by the SAS system indicate that the current active DOS directory where SAS is invoked plays an important role in what function key settings and screen attribute settings will be used during the SAS session.

From the system administrator standpoint, this process should be monitored to save disk space, especially if changing the default settings is not an action that is likely to be taken by the users.

From an application development standpoint, these actions by the SAS system are very useful, essential, and should be used to improve the functionality of the application. It is logical that every application should use one function key settings and screen attribute settings to make it convenient for the user to follow. It is also conceivable that the application developers tailor the default settings to suit certain applications or client requirements and expect the SAS system to be able to find those tailored settings and use them. This goal can be accomplished by changing the default settings, storing them in a user profile catalog, storing the catalog in a subdirectory called SASUSER, and finally, keeping this SASUSER subdirectory in a private directory dedicated for the application. More information about this process will be presented in topic number 8 (Integrating all previous pieces to work as one system).

Finally, we should become familiar with the reserved file name extensions used by the SAS system. The SAS system automatically supplies the file name extension for system files. Some of those file extensions are:

- .EXE - used for executable SAS system files
- .MSG - used for SAS message files
- .SSD - used for SAS data sets
- .SCT - used for SAS catalogs

In addition to the familiarity with the SAS system file and subdirectory structure, it is also useful to briefly overview SAS data libraries and catalogs. Figure 1 illustrates the relationship between SAS data library, SAS files, and their elements.

![Figure 1: SAS Data Library Structure](image)

SAS data libraries are used to store SAS data sets and SAS catalogs. This concept provides a means to keep the application files organized. It is important to note, however, that this is only a logical concept and not a physical collection. In other words, defining a SAS data library, by using the LIBNAME statement, merely establishes a path to a DOS subdirectory. The SAS system does not keep a list of files or SAS data sets in the DOS subdirectory. Other DOS files could reside in the subdirectory. Defining a SAS data library, however, establishes a means by which we can logically group SAS data sets in one place and call it a library. SAS catalogs store several types of entries. Those entries are used by several SAS products such as SAS/AF and SAS/FSP. The following is a list of the different entry types that can be stored in any SAS catalog:

- AFMACRO
- CMAP
- FORM
- GRSEG
- INFMT
- FFMTC
- LETTER
- MENU
- PLOPTION
- TEMPLATE
- AFMASTRO
- BTRREE
- CONVT
- FORMAT
- HELP
- IMOD
- KEYS
- LIST
- MATRIX
- NPAD
- PROGRAM
- SCREEN
- WSAVE

4. FILE STRUCTURE AND NAMING CONVENTION FOR YOUR APPLICATION

Every software development company and group adopts a slightly different approach in their practice. They assign different priorities to the same topic. Some believe in doing things correct the first time, others believe in experimenting. Many groups believe that if a recently
developed module works the first time, it will always work. Few, on the other hand, strongly believe that spending the time to perform thorough testing pays off at the end. The real test of all these approaches is the process of software application development.

Two of the most significant requirements for successful development are a good file structure and a meaningful file naming convention. This issue becomes even more vital when developing a large application. Designers have to spend time at the beginning of the development process to design a file structure for the application. A good file structure should have the following features:

- **Comprehensive:**
  It must satisfy all systems requirements.

- **Flexible:**
  It can be easily expanded to accommodate future enhancements and additions to the system.

- **Manageable:**
  Can be implemented within the given hardware and time frame.

- **Maintainable:**
  Easy to maintain given the capabilities of the client (system users).

File naming convention is another important topic that makes a noticeable difference during the development, documentation, installation, and system support phases. A file naming convention must cover all possible file types and variations. The following is just an example of a file naming convention used in developing a PC-based system called "ACAP" (Army Career and Alumni Program) using the SAS system version 6.04. The ACAP system is used to help Army personnel transition out of the service.

```plaintext
C:\ACAP
  - catlib <dir> subdirectory
  - datalib <dir> subdirectory
  - programe <dir> subdirectory
  - macros <dir> subdirectory
  - frmtlib <dir> subdirectory
  - ssasuser <dir> subdirectory
  - ssaswork <dir> subdirectory

In this file naming convention, all the files and programs which belong to the ACAP system are grouped under one main subdirectory "C:\ACAP." SASUSER and SASWORK subdirectories are included in this main subdirectory. The "C:\ACAP\CATLIB" represents a SAS catalog library, where two catalogs, ACAPAF.SCT and ACAPFSP.SCT, are stored. All SAS/AF screens are stored in the ACAPAF.SCT catalog, whereas all SAS/FSP screens are stored in the ACAPFSP.SCT catalog. The "C:\ACAP\DATA\AL\" subdirectory contains all ACAP system files. The "C:\ACAP\PROGRAMS\" subdirectory is used to store all SAS programs in external files with a "PGM" extension. The "C:\ACAP\MACROS\" is used to store SAS macros in external files with a "MAC" extension. And finally, the "C:\ACAP\FRMTLIB\" subdirectory is used to store the user defined format catalog. The names of the external files in the "C:\ACAP\PROGRAMS\" and the "C:\ACAP\MACROS\" subdirectories, and the names of all AF screens and FSP screens start with one of five letters indicating the five main modules in the ACAP system. For instance, the external file "PADDPRNLPGM" indicates that it contains a SAS program that is used in the Personnel Module (starts with the letter P). The program is used in the process of adding a new personnel record.

This type of organization allows the developers to take advantage of file caching. A concept that would improve the speed of accessing files. The important point to emphasize is that file caching is useful for code (programs or macros) and not for SAS data sets. By storing the code in a separate subdirectory, it will be possible to use disk caching for this subdirectory. Utilizing file caching would not have been possible if the code and the SAS data sets were stored in the same subdirectory.

5. **OVERVIEW OF ESSENTIAL SAS STATEMENTS AND SCL FUNCTIONS USED IN ANY APPLICATION**

This topic will overview some of the most essential SAS statements, LIBNAME and FILENAME. It is almost impossible to develop an application without using these two statements. These statements are usually executed at the beginning of the application. They associate short names, referred to as library reference and file reference, with a library and an external file.

The SAS LIBNAME statement associates a libref (a word up to eight characters long) with a DOS subdirectory. For instance, the statement:

```
LIBNAME CATLIB 'C:\ACAP\CATLIB';
```

associates the libref CATLIB with the DOS subdirectory C:\ACAP\CATLIB.

Similarly, the SAS FILENAME statement associates a fileref (a word up to eight characters long) with an external DOS file. Defining the external file should include the directory path plus full file name. For instance, the statement:

```
FILENAME PADDPRNL
  'C:\ACAP\PROGRAMS\PADDPRNLPGM';
```

associates the fileref PADDPRNL with the external file PADDPRNLPGM that is stored in the subdirectory C:\ACAP\PROGRAMS.
File references are used to refer to external files in SAS programming statements such as INFILE, FILE, and INCLUDE statements.

The effect of LIBNAME and FILENAME statements remains during the entire SAS session. The two statements can be used anywhere in the SAS code.

Some of the useful SCL functions are EXIST, CEXIST, and FEXIST. They perform similar tasks. The three functions enable us to ask the SAS system to search a specific DOS subdirectory for the existence of either SAS data sets, SAS catalogs, or an external file. The three functions return a code of 1 or 0, to indicate the existence or nonexistence of the object. The following is an example of the exist function to search for the existence of the SERVICES data set in "C:\ACAP\DATA\LIB" subdirectory:

LIBNAME DATA\LIB "C:\ACAP\DATA\LIB" ;
RC = EXIST (DATA\LIB, SERVICES) ;

The following is an example to show the use of the CEXIST function to search for the ACAPAF catalog in the subdirectory "C:\ACAP\CATLIB"

LIBNAME CATLIB "C:\ACAP\CATLIB" ;
RC = CEXIST (CATLIB, ACAPAF) ;

Finally, a similar example to show the use of the FEXIST function:

FILENAME PADDPRNL
"C:\ACAP\PROGRAMS\PADDPRNL.PGM" ;
RC = FEXIST (PADDPRNL) ;

In an application development effort, it is very likely to have a need to create permanent SAS data sets. For instance, if an application allows the user to enter data about several projects, then we must have a mechanism to search a DOS subdirectory for SAS data sets containing data about those projects. If such SAS data sets exist, then we will allow the user to append new data to existing data in the proper data set. Using the EXIST function will help in performing such a task.

6. WHERE DOES THE OUTPUT GO AND WHAT CAN WE DO ABOUT IT

The SAS execution mode, for instance, batch mode or interactive mode, determines where the procedure output is routed. In the mainframe environment we usually get a hard copy of our batch job output. During a display manager session in the mainframe, the SAS system routes the output to the display manager output window as a default action. There is always a default action taken by the SAS system. What happens in the PC environment? Can application developers control the routing process? This topic will attempt to answer these questions.

When we run a SAS batch job in the PC environment, the SAS system creates two DOS files with the same name as the DOS file, containing the SAS program but with different extensions. The log resulting from executing the SAS program, will be routed to a DOS file with the extension LOG. The output resulting from executing the SAS program will be routed to a DOS file with the extension LST.

From an interactive application development standpoint, the SAS system will be operating in an interactive mode. Assuming that SAS/AF product is used in the development, we need to be able to control the process of routing the output. Generally, when a SAS/AF screen or SAS/FSEDIT screen is active, the display manager will also be running behind it. In almost every application, there is some type of output. Some applications display the output to the user, some route it to an external file, and others route the output to a printer.

The default action taken by the SAS system is to route the output to the display manager output window. If the user requested a report, for instance, by selecting an option from an AF menu, the SCL code should submit the SAS code to produce the report. However, the report will be routed to the display manager output window which is behind the AF menu but not active. As a result, the user will not be able to view the report. What can we do to route the output to a printer or to save it to an external file?

The following is one suggestion to answer these questions:

- To make the display manager output window active, include a call to the EXECMD routine in your SCL code after executing the SAS code that will produce the report. The following call will make the output window active. You can then instruct the users to type AF at the command line to go back to AF.

CALL EXECMD (OUTPUT; ZOOM; TOP);

First we go to the output window, then we zoom to occupy the entire screen, and finally go to the top of the report.

- To route the output to a printer, we can use PROC PRINTTO. By using this procedure, we will be able to redirect all procedure outputs to a new destination. The following code defines a local printer, routes the output to the local printer, and finally directs the output back to the default destination.

FILENAME LOCALPRN PRINTER 'PRNT' ;
PROC PRINTTO PRINT = LOCALPRN ;
RUN;
To route the output to an external file, a very similar set of SAS statements will be used. First, define the external DOS file called REPORT.PRN, then route the output to it, and finally direct the output back to the default destination.

FILENAME RPTFILE
'C:VACAPREPORT.PRN' ;
PROC PRINTTO PRINT - RPTFILE;
RUN;
/*  SAS code to produce the output */
PROC PRINTTO;
RUN;

SAS - DMS

This command will invoke a SAS display manager session.

The CONFIG.SAS file is a DOS file that stores all the SAS configuration options. When we issue the SAS command, the SAS system locates CONFIG.SAS and reads the options from it. Here are only some of the configuration options that can be stored in CONFIG.SAS:

-AUTOEXEC specifies the name of the AUTOEXEC file that SAS should use
-DMS specifies an interactive SAS display manager session
-EMS specifies the amount of expanded memory to be used by SAS
-FILEBUFFERS specifies the number and size of file buffers we want SAS to use
-FILECACHE specifies the name of the subdirectory that can be cached and the number of files from this subdirectory that can be cached
-FSDEVICE specifies the monitor device driver

From an application development standpoint, this file should be edited to adjust the configuration options as desired. The most important option that should be set is the AUTOEXEC option. This option should point to a special AUTOEXEC file to be used by the SAS system for the particular application that is being developed.

The AUTOEXEC.SAS is another important DOS file that SAS looks for when it is invoked. This file can contain any SAS statements. The SAS statements are executed automatically every time the SAS system is invoked. The AUTOEXEC file provides an ideal place to perform the following tasks:

1. Issue all the LIBNAME statements used by the application;
2. Issue all the FILENAME statements used by the application;
3. Issue an OPTION statement to set SAS system options to meet the application requirements;
4. Define application Global macro variables;
5. Initialize Global macro variables;
6. Set options and initialize global macro variables to match the specific requirements for the client(s) site;

7. IMPORTANT DOS FILES (CONFIG.SAS and AUTOEXEC.SAS) AND HOW CAN WE USE THEM ADVANTAGEOUSLY

This topic will include an overview of two important DOS files, the CONFIG.SAS and AUTOEXEC.SAS. Before discussing these two files, we should very briefly review a more general topic, the SAS global options. The SAS global options consist of two types, configuration options, and SAS system options.

Configuration options generally affect the features of the SAS system and the way the SAS system interfaces with the hardware and the operating system. Because of the nature of those options, they have to be set when we invoke the SAS system. Once those options are set, they can not be changed during the SAS session. Those are the kinds of options that are included in the DOS file CONFIG.SAS. To check the current configuration options, invoke SAS using the option VERBOSE as follows:

-SAS - VERBOSE

A list of the current configuration will be displayed on the screen.

The second kind of SAS global options is the SAS system options. Those are the kinds of options that affect, for instance, the number of observations to be processed and the appearance of SAS output. Those options can be set dynamically as many times as we want during the SAS session.

Specifying or resetting configuration options can be done in two ways. We can edit the DOS file CONFIG.SAS and change the options, or we can specify the options with the SAS command as follows:

FILENAME RPTFILE
'C:VACAPREPORT.PRN' ;
PROC PRINTTO PRINT - RPTFILE;
RUN;
/*  SAS code to produce the output */
PROC PRINTTO;
RUN;
7. Start the application. For instance, by calling a Welcome screen or a Main Menu.

To elaborate on task number 6, here is an example:

A global macro variable called DRV may be used throughout the application to indicate the name of the hard disk drive where the application resides. It is very possible that one client wishes to install the application in hard disk drive D, while another client prefers to install the application in hard disk drive F. Throughout the application, the developers should refer to the programs, SAS catalogs, and libraries using the macro reference &DRV. For instance, library names should be coded as follows:

LIBNAME DATALIB "&DRV\ACAP\DATALIB" ;

In the AUTOEXEC file, we define DRV as a global macro variable and assign it the name of the client hard disk drive as follows:

%GLOBAL DRV ;
%LET DRV = C: ;

8. INTEGRATING ALL PREVIOUS PIECES TO WORK AS ONE SYSTEM

Now that we have covered all key topics, we need to review a mechanism by which we can combine all these elements together as one system. This topic will present one of many ways to accomplish this task. The ACAP system will be used again to illustrate this topic. The ACAP system is a PC application used to help Army personnel transition out of the service.

First we need to create a DOS main subdirectory and call it C:\ACAP. This main subdirectory will have the following subdirectories: CATLIB to store SAS/AF and SAS/FSP catalogs; DATALIB to store all the SAS data sets used by the ACAP system; PROGRAMS to store all SAS programs; MACROS to store all SAS macros; FRMTLIB to store a format library catalog.

In addition to these subdirectories, we will also have SASWORK and SASUSER subdirectories for the SAS system to use. Two DOS files, AUTOEXEC.ACP and CONFIG.ACP will also reside in C:\ACAP main subdirectory. This DOS subdirectory structure will look like the following:

```
C:\ACAP
  CATLIB <dir> subdirectory
  DATALIB <dir> subdirectory
  PROGRAMS <dir> subdirectory
  MACROS <dir> subdirectory
  FRMTLIB <dir> subdirectory
  SASUSER <dir> subdirectory
  SASWORK <dir> subdirectory
```

There is a probability of uploading this system to a mainframe TSO environment, all the SAS macros are separated from the SAS programs and stored in the MACROS subdirectory. This will facilitate uploading all macros to a TSO Partitioned Data Set and take advantage of the MACRO AUTOCALL feature of the SAS 6.06 system in the mainframe.

A meaningful file naming convention is implemented to indicate the contents of every file and to facilitate the documentation process.

During the development process, SAS is invoked from the C:\ACAP subdirectory, and while in display manager, the function keys are reset to meet the requirements of the system. Also during the process of building FSEDIT screens and AF screens, function keys and other screen attributes are reset. The SAS system will save the new settings to the user profile stored in C:\ACAP\SASUSER.

The DOS file CONFIG.ACP is a copy of the CONFIG.SAS provided by SAS Institute. CONFIG.ACP is then edited to adjust some configuration options such as:

```
-AUTOEXEC C:\ACAP\AUTOEXEC.ACP to instruct the SAS system to use the C:\ACAP\AUTOEXEC.ACP file as the autoexec file, and -EMS ALL to instruct the SAS system to use all the available expanded memory.
```

The DOS file AUTOEXEC.ACP has various SAS statements such as an OPTION statement to set SAS system options, all the LIBNAME and FILENAME statements, a %GLOBAL statement to define all global macro variables, %LET statements to initialize global macro variables, and finally a call to the first AF welcome screen to start the ACAP system.

Finally, ACAP.BAT, a DOS batch file, is created. This file resides in the DOS root directory and will perform the following:

- Change directory to C:\ACAP
- Invoke the SAS system with the option -CONFIC C:\ACAP\CONFIG.ACP to instruct the SAS system to use C:\ACAP\CONFIG.ACP as the CONFIG file and to read and use our specific system configuration options from it.
- Change directory back to the DOS root directory after the ACAP system session is terminated.

With this mechanism, the user will not be concerned about where the ACAP system resides and how to get there to invoke it. Also other SAS users will not accidentally use or modify the specific system configuration options or user profile which contains tailored function keys settings.
CONCLUSION

This tutorial discussed some of the essential and common topics to the process of application development using the SAS system version 6.04 in the PC environment. Although there are several approaches and philosophies to this dynamic process, application developers will find these topics common to all of them. The introduction of the Screen Control Language combined with the enhancements to the SAS/AF and SAS/FSP software facilitate the software development process, and offer a tremendous amount of flexibility and features. The use of the SAS macro language in conjunction with SAS/AF software is inevitable. Macro variables represent the vehicle by which information is passed between SAS/AF screens.

Familiarity with the SAS system files and subdirectories is essential to effectively use and maintain the SAS system in the PC environments. Application developers should take advantage of the important DOS files, CONFIG.SAS and AUTOEXEC.SAS. These two DOS files contain significant information to the SAS system, and can be tailored to satisfy the requirements of any specific SAS application.

The SAS system routes the procedure output to the display manager output window. Using the SCL EXECMD routine, FILENAME statements and PROC PRINTTO enable us to redirect the output to a local printer or to an external DOS file. A successful PC-based application begins with a comprehensive file structure and a meaningful file naming convention, and continues to be the most challenging and rewarding task in the Data Processing career.

REFERENCES:

7. SAS Institute, Inc. Installation Instructions The SAS System for Personal Computers: Release 6.03; Cary, NC; SAS Institute, Inc.

*SAS is a registered trademark of SAS Institute, Inc.

The author can be contacted at:

HAY Systems, Inc.
2000 M Street, N.W.
Suite 650
Washington, D.C. 20036

Telephone: (202) 223-3703
FAX: (202) 293-5369

315