

**Creating a PC-based Information System:  
An Evolutionary/Revolutionary Visionary Approach  
(Part II)**

Peter Parker, U.S.  
Dept. of Commerce,  
Washington D.C.

R. Keith Daly, U.S.  
Dept. of Commerce,  
Washington D.C.

7/02/1997

**ABSTRACT**

This paper is intended for former mainframe users who now use PCs for their data processing. It is a continuation of a paper we presented at SUGI 21, which described our switch to the PC platform, the considerations we made, and the problems we encountered. Here's what we've done since then- especially new innovations initiated and problems encountered.

The Office of Textiles and Apparel (OTEXA) of the U.S. Department of Commerce has moved all databases and all computer programs from the NIH mainframe computer system onto a PC based system. Details on this transition can be found in our paper in the SUGI 21 proceedings.

Since we've developed this PC-based system, we have become more proficient in writing more:

- efficient code (using indexes and advanced data step statements).
- elaborate SAS macros and macro windows to automate production and data queries.
- elegant applications that tie SAS software programs with other software applications.

As well, this paper will elaborate on problems encountered and ideas that are being pursued.

We use the OS/2 and Microsoft Windows 6.12 versions of SAS software, along with most commonly used PC software on a quad Pentium Pro, dual Pentium and Pentium Pro PCs with multi-gigabyte hard drives and multiple CD-ROM units.

**INTRODUCTION**

The U. S. Textile Program of the Dept. of Commerce is concerned with the health of the domestic textile and apparel industry. The program monitors and establishes import agreements with other countries, assesses the

domestic industry, and promotes market expansion. The evaluation of the U.S. textile and apparel markets and imports from all countries, category by category, allows for the identification of potential market disruption here in the United States. Analyses of the domestic industry may determine that market disruption or the threat of disruption exists. Such a conclusion may lead to the establishment of import quotas for specific commodities and countries.

The Textile Information Management System (TIMS) began in 1978 by congressional mandate. TIMS produces all of the Textile Program's periodic reports on time, and maintain an easily accessible collection of databases for time-critical, situation-specific query and analysis. About forty on-site Commerce international trade professionals and economists view these reports on their PCs via a recently implemented object-oriented software application on the LAN and on CD/R discs. Currently, this information is being processed on a quad Pentium Pro File Server running OS/2 Warp Server SMP. TIMS is implemented by two computer programmer analysts (the authors) and a computer assistant.

We operate out of a studio/laboratory/workshop environment, dabbling with and researching various hardware and software technologies, designing data processing systems, writing code, pulling chips out of mother boards, assembling PCs, searching computer science information bases, and envisioning the future of information technology. SAS software is used to create and maintain the import and export databases and to produce the TIMS reports and data tables. Most of these reports, some of them numbering hundreds of pages, are generated monthly and a few weekly. Ad hoc reports ranging from a few pages to a few hundred are generated on a daily basis for OTEXA professional staff.

For more details on how we became a PC-based data processing shop, see our similarly named paper (i.e., Part I) in the SUGI 21 proceedings.

**WHAT WE HAVE DONE SINCE SUGI 21**

Since we've developed a PC-based system last year, we have become more proficient in writing more:

- efficient code (using indexes and more complicated data step statements)
- elaborate SAS macros and macro windows to automate production and data queries (and some non-SAS macros)
- elegant applications that tie SAS software programs with other software applications

Of course, as with any newly-created complicated data processing systems, we have encountered a few problems. After solving these problems, we will anticipate the future (vision), enhancing our prototype computer system.

**Efficient Code-**

During the years of using exclusively mainframes, code efficiency was not a major

concern for us, since the NIH Computer Center's raw power could more than compensate for any sloppy coding and because most big jobs were run overnight. However, as we moved applications to a 486 PC (then to a dual Pentium and later to our present quad Pentium Pro PC), we no longer had that luxury. Having less horsepower made us better programmers. We had to tighten up our code or processing our applications would take too long. In our information environment, analysts need their data reports immediately. Obvious efficiency steps include judicious use of keep and drop statements to limit the sizes of these data sets. Additionally, removing superfluous code such as unneeded sorts was helpful. Writing more complicated SAS data steps is another way to speed up our processing, specifically more intensive uses of @ (input record holds) in the input statement, where clauses (after indexing data steps), creating generic code to read any dBase file and using dynamic data exchanges to modify Quattro Pro spreadsheets.

When we read U.S. export data, involving millions of records with about 200 characters each, we need to extract out those with specific Textiles and Apparel commodity codes. This Data step may take up to four hours to process on a dual Pentium PC. By using the @ function, we read in only part of the record to see if the schedule B value is in the right range. If it is, then we read and output the entire record. This simple coding change alone dropped this Data step from four hours to about one hour.

When we create permanent SAS data sets, we now index them. Most of our queries involve particular countries. On the mainframe, we made two types of import data sets. One would contain the imports for a month, the other for a particular country for a year. Thus, if one wanted to see all of China's imports for a year, it would not be necessary to read sequentially twelve months of data for all countries, but only one data set for China. However, on the PC, we create only the monthly import data sets but indexed on the country variable, saving a considerable amount of storage space. For a year's worth of China imports, we still read twelve large import data sets, but not sequentially. The time required for our data queries have dropped from minutes to seconds. When running several data queries, the timesaving will be significant and throughput speedier.

Another speed enhancement, unavailable to us in a batch-driven mainframe environment, was running SAS interactively. Rather than scheduling a SAS job, we were now assured of being on top of the job input queue since we're running most of our computer programs from our own desktops, using our OS/2 Warp PC only as a file server. Even better, we could run any portion of a job at will, at a substantial time-savings. For instance, we could submit a job that filters out gigabytes of data, manipulates the extracted data, and then prints out reports from it. It may take thirty minutes or more to extract the data needed into a temporary SAS work data set. After that, additional code could be submitted from the clipboard, using various procs. This is particularly useful if a mistake is made in the code after reading all of the input data. Instead of resubmitting the

entire job, only the part after the corrected code needs to be.

Another enhancement was setting up generic code to read any dBase file. Using the @ function, we read information off the first record of a dBase file and use that to generate the input statement with SAS variable names derived from the dBase fields. This step makes it easier to maintain non-SAS data sets and their associated programs. If we change the dBase field layouts, we don't have to change the SAS programs. As well, we skip converting the dBase files to ASCII text before reading it using our SAS programs. We have written code to write SAS data sets to dBase files directly within the program, reading information off of the Proc Contents of a SAS data set.

Another technique to interact with non-SAS software is to use Dynamic Data Exchanges (DDE). Using this advanced SAS feature, we can write SAS programs to update Quattro Pro spreadsheets. Often analysts need filtered SAS datasets in a spreadsheet format so that they can manipulate the data themselves on the fly. Often they'll need to calculate new fields such as percent changes subtotals or to create line graphs on time-series data.

#### SAS Macros and Macro Windows-

On a mainframe, we have used Wylbur (a mainframe type of text editor) to open up SAS program templates, edit and then submit them for processing. This handling of programs incurs the risk of accidentally saving these temporary modifications to the original programs. As we become more sophisticated programmers, we set up simple runstreams that queried us for parameters and then submitted these jobs in batch mode. On the PC, using macros, macro windows and other features of OS/2 SAS, we can set up more comprehensive techniques to process monthly and weekly production and create queried reports, with little chance of the kinds of error you can make when working hastily.

Using macros and macro windows, we have done the following:

1. **Created program icons** that the user can click on to submit a job (increasing ease of use).
2. When one clicks on these icons, several macro windows appear, **prompting the user for parameters** like time periods, country names, commodity codes, and type of queries (decreasing human error).
3. Since these icons can be run on the LAN, the first macro window asks for the user's name. Based on the user name, the libname and filename drive designations (macro variables) are mapped. That is, for person A, the import data sets might be on their S: drive, while for person B, it might be their Z: drive on the LAN. The macros in the programs **take care of this housekeeping**.
4. On the mainframe, the user may save copies of their programs with their particular parameters into another file to run later. This step avoids the nuisance of retyping complicated queries such as those requiring several dozen 10

digit commodity codes. Since only the macro field values are saved, reruns always use the most current version of these programs. When rerunning these programs, one still sees the same Macro windows, but the parameter are already filled in as a default. Of course, the user can change those values, as necessary.

5. One drawback of Macro windowed code is that the user has less control in modifying the program, subject to what macro variables appear on the screen. To most users, this is not a hindrance. However, to advanced users who know SAS coding and want to make major changes, this could prove to be unworkable. We added an option to add up to ten lines of code at the end of the first data step where data is extracted based on these parameters values. This additional window should take care of most customized situations. As well, a macro window was developed to allow the user to determine what variables will be used for report sorts and control breaks. Making these programs flexible in spite of the rigidity of macro windows is an ongoing process.

6. Within OS/2 SAS are external references to non-SAS code. If one wanted to read a 9-track tape data set in monthly production, one would have to use special tape software to copy the file to an ASCII file. One would have to set up the software for the lrecl, blocksize, and the conversion parameter, the name of the output ASCII file and its correct drive. Going through these many steps easily could lead to many user errors. Using macros and the X statement, the tapes are automatically copied to the right place in the right format when the SAS program icon is executed. Our production coordinator only has to load the correct 9-track tape.

7. Using the %include macro, we save chunks of common code in a library directory. This feature allows us to change some code in one place, rather than in several different programs.

#### **Non-SAS macros -**

After successfully using SAS macros, we have begun to design macros in other Software applications. One applications is create a button on a Quattro Pro 6.0 tool bar that enables the user to take blocked cells and save them as an ASCII file, selecting from a pick list of different names. These files are included with SAS generated reports as footnotes, viewed through an application developed by object oriented software. We re currently working on similar macros in WordPerfect 6.1. All of these macros have the same advantage, combining several keystrokes into a click of a button or icon and then selecting parameter values.

#### **More elegant applications**

##### **Data Reports on LAN and CD/R Discs**

Since we have left the mainframe environment, we have devised better ways of querying and viewing our databases. Not having the large scale printers of mainframe systems forced us to give up paper. Instead of distributing reports on paper, some numbering hundreds of pages, we have developed Intranet and CD-ROM disc alternatives.

On the LAN, the analysts can view all of our reports by clicking on an icon. This click starts up object-oriented applications, written using Delphi, Visual Basic and Visual dBase software. The user can now view the reports in exactly the format as the former hard copies, but with the advantage of seeing different pages faster than flipping through paper and having data query search capabilities.

We have put all of our published reports on CD-R disc, using the same software. We could have created the same presentation software using SAS-AF software, but this idea would be infeasible, because of the problems of bundling SAS executables on these CD-ROM discs. We understand that the SAS Institute is working on this problem, primarily from a legal standpoint on licenses. On paper-published reports, we process a lot of forms and then wait until the data press release date before sending them to the printers. After a few weeks, we will receive several boxes of publications and then send them out to our customers. Now, we produce CD-R discs on our CD-R duplicator and start sending them out on the press release date, selling them for a small fraction of the hard copy price and requiring considerably less paper work. As well, our analysts frequently travel and they now can carry all of our publications in a CD-R disc, rather than a large stack of bulky paper reports.

#### **Internet Applications**

We have created an OTEXA homepage and put some of our most important publications on it, in addition to selling them on CD/R discs. Since most of our files on our homepage involve several files with HTML formatted data, we have written SAS programs with clever macros to convert parts of our SAS data sets to ASCII with the HTML codes included. Updating our homepage involves mostly running a few SAS programs and then copying these SAS-generated HTML files to our Internet server.

#### **Problems Encountered**

1. Macro debugging- While macros are extremely useful, they also make the computer programs much more complicated. In order to make the programs more readable, we have subdivided them into different macros as modules. The final module is the driver module, which executes the previous macros, based on macro variable values. This style is similar to structured top-down design used by COBOL programmers. The other problem with macro-laden programs is that the SAS log does not produce line numbers for code within macros. This makes finding errors more difficult since the line referenced in an error may in fact be several lines of code.

2. Printing- Printing in most software packages seems to be a problem, especially in SAS applications. Sometimes it s a trial and error process. Also, Proc Freq tables print only one per page, even when setting the SAS option for huge paper sizes. The printing problems are becoming less significant as reports are appearing on screens rather than paper.

3. Encouraging analysts to shift their work paradigms- No matter how easy we can make the screen queries, some analysts still prefer their

reports on paper. It s more portable for them and they can easily mark them with hand-written notes. We made printing a feature available on their Object-Oriented viewing applications.

4. Persuading analysts to use their mice- most of these PC applications require some manual dexterity from the users- pointing, blocking and clicking with their mice. Some users will have problems with that and a systems analyst should anticipate that.

#### **Future Directions** (Work in Progress)

1. Setting up system where analysts can electronically submit queries (rather than on paper) which automatically run in SAS and then informs the analyst when it is complete.
2. Setting up Internet applications that interact with our SAS databases directly, instead of the snapshots of them in HTML format.
3. Hardware improvements:
  - A. More powerful PCs(beyond quad Pentium Pros) and more powerful software(OS/2 Warp Server4.0)
  - B. Auto-load printer labels for CD/R discs (now we have auto-load capability for only producing CD/r discs).
  - C. More hard disk space (you never have enough).
  - D. Faster and better CD/r drives (12x have already come out and DVD drives will soon follow).

#### **In Conclusion**

Since we left the mainframe environment to a PC-based one, we have been exploring territory new for everyone. Necessity (a PC will never be the equal of a mainframe in raw power) has forced us to become better programmers, squeezing out as much efficiency as possible out of our code. We have taken advantage of technological opportunities not found in the relatively static world of mainframe computing. We have developed new ways of viewing data, using object oriented software, CD-ROM disks, and internet/intranet software. Paper has become obsolete.

We anticipate that technology will continue to progress at this rapid rate, and we plan to incorporate new computer inventions as we maintain and upgrade our PC-based system. By next year, we should have enough innovations to warrant presenting Part III of Creating a PC-Based System-An Evolutionary/ Revolutionary/ Visionary Approach, SAS on the Web.

#### **ACKNOWLEDGMENTS**

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. IBM and OS/2 are registered trademarks or trademarks of International Business Machine Corporation. Indicates USA registration.

Other brand and product names are registered

trademarks or trademarks of their respective companies.

Peter Parker  
R. Keith Daly  
U.S. Dept. of Commerce  
Room 3100, Main Commerce Building  
14th & Constitution Ave., NW  
Washington, D.C. 20230  
(202) 482-1449  
Peter\_Parker@ita.doc.gov  
Keith\_Daly@ita.doc.gov