Supporting the "Program-Analyze-Write-Review" Process with a Development Environment for Base SAS® and the Macro Language

Barry R. Cohen, Planning Data Systems, Inc.

ABSTRACT

Pharmaceutical companies do much Base SAS and Macro Language programming in the process of analyzing their clinical trials data. This is one part in their larger process of collecting, managing, analyzing, and presenting this data. Many companies now provide considerable system support for the early part of this process (data collection and management) and for the later part (document publishing and presentation of results). But less has been done to support the analysis part in between; i.e., the process of developing and executing SAS programs for statistical analysis, and then writing and reviewing an analysis document based upon the results of program execution. This "Program-Analyze-Write-Review" process is a major, time-consuming process in any research organization, and I address automated support for it in this paper. After a top-level discussion of the full Program-Analyze-Write-Review process, I focus in more detail on support for the SAS program development component within this process.

INTRODUCTION

As pharmaceutical companies conduct clinical trials, they generally follow the process presented in Table 1 regarding the data and analysis:

Process Activity	Major System Support
Prepare, manage data	yes
Develop analysis programs	no
Run programs, analyze data	no
Write analysis document	no
Review document	no
Publish document	yes
Present document, data, programs	yes

Table 1: Clinical Trial Process Regarding Data and Analysis

Much attention has been given in this industry to system support for data preparation and management, for document publication, and for document and data presentation. For example:

- Large-scale Clinical Trials Systems have been and are being developed to handle clinical data collection, preparation, and management.
- Document Management Systems are being used to publish New Drug Application (NDA) documents.
- CANDAs (Computer Assisted New Drug Applications) are being used to present the documents, statistical analysis, and data.

But perhaps surprisingly the middle of this process, i.e., the program, analyze, write, and review steps, (shaded in Table 1) has received less system support. I will refer to this middle part as the PAWR process (from **P**rogram, **A**nalyze, **W**rite, **R**eview). So, for example, often:

- SAS programmers are on their own to develop programs. No facilitating environment beyond SAS Display Manager is available to support them as they code, test, debug, store, and retrieve the generations of their programs.
- Statisticians are on their own to analyze data and write the statistical portion of NDA documents. No facilitating environment is available for them to retrieve and execute the generations of SAS programs, retrieve and review the program outputs matched to these generations, and move the program outputs (tables and graphs) into the analysis document they are writing. Clinical staff are similarly on their own to analyze data and write the medical portion of analysis documents.
- Statisticians and clinical staff are on their own to review the document. No facilitating environment is available for them to access the written document, to annotate and then share their review comments in the document, to move from the document text and tables to view the particular patient group data behind them, to re-create a table via changes to program parameter settings in order to explore alternative approaches to the analysis, or to view additional data about the patient group involved in the table.

The limited amount of integrated system support for the PAWR process is somewhat surprising because the PAWR process is a major part of the larger clinical trials process, and effective system support could have a commensurate major impact. But I do see two

possible reasons why such support has not yet been forthcoming. First, it requires integrating activities that are disparate from the information technology perspective. Specifically, the PAWR process is part software-development oriented, part document-development oriented, (and even has a data handling component). Second, each activity has received at least some system support *within that activity*, which has allowed the industry to reach an acceptable level of functioning by tying these activities together manually, while putting their system development resources instead into more pressing needs.

But I feel that an organization can get much benefit from providing integrated system support for the PAWR process. I also feel that now is a good time to do so because, for many organizations, the more pressing needs of the greater clinical trials process have been or are now being addressed. Also, I feel that more recent advances in the IT industry now make it more feasible to integrate these technically disparate activities. (I am thinking specifically of the ability to tie together a document environment and a program/data environment).

This, then, is the subject of my paper. I will present some thoughts about the activities of the PAWR process --- thoughts that describe the process and are a necessary precursor to building an effective software application to integrate and support the activities of this process. I will also present some thoughts about what such a software application might look like, at least in part. But even more basically, and perhaps more importantly for now, I will simply talk about the PAWR process as a process. We cannot effectively support a process if we do not readily see it as a process, as a business cycle with a cycle time, and with activities within it where time can be saved if appropriate support is provided.

I hope to foster more dialog within the SAS community on this subject through this paper. Hopefully, we can collectively generate ideas that might lead to additions to the SAS System in this regard and to development of our own applications in this area. But I do feel that since so much of the SAS System is about analysis, that if we generate information that describes the activities of the analysis process we conduct, and describes where these activities are and are not currently supported from a system viewpoint, then we will probably be generating information which will eventually be reflected in new features of the SAS System.

PROGRAM-ANALYZE-WRITE-REVIEW PROCESS

The activities of the PAWR process and their flow are illustrated in Figure 1. The process begins in the lower left of the diagram with the Program Development Environment (PDE), where Base SAS/Macro Language programs are developed and executed. The activities in the PDE include writing, testing, and debugging code, and eventually executing the code in production to produce the tabular and graphic outputs which express the analysis in raw form and which will become part of the analysis document.

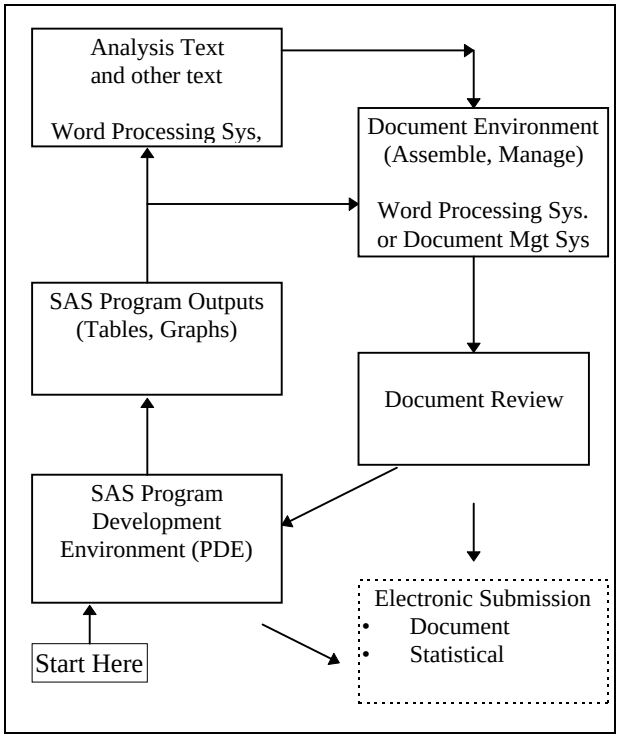

Figure 1: Program-Analyze-Write-Review Process

The process continues with the analyst writing the text which discusses the results of the analysis, referring to the tables and graphs. This typically occurs in a word processing environment. Other text is typically also written, generally before the analysis even occurs. The analysis text, other text, and the tables and graphs are next assembled and managed. If the word processing environment is used for this function, the tables and graphs, which are SAS outputs, are incorporated into the word processing environment. Alternatively, a document management system, (a different genre than word processing software), may be the place where the text and the SAS outputs are assembled and managed. If so, the SAS outputs must be loaded into this software-controlled environment instead.

The next activity of the PAWR process concerns review of the document. Typically in the pharmaceutical industry many people beyond the

author are involved in the review. This activity includes feedback and questions to the authors from the reviewers, and can also involve revising and re-running programs to re-produce SAS outputs and re-write analysis text, which then must be re-loaded to the word processing environment or document management environment.

Finally, the dotted-line box in the lower right of Figure 1 indicates that once the analysis is complete and the document is complete and published, it is often submitted electronically (as well as in hard copy) to a regulatory agency for review. This is sometimes referred to as a CANDA, for Computer Assisted New Drug Application. And the statistical analysis (programs and data) may be submitted in a Statistical CANDA, too. I show this activity in a dotted-line because some (many?) companies do not treat it as part of the PAWR process, but as more of a follow-on activity.

PAWR Activities Un-integrated

From the software application point of view, in the pharmaceutical industry today, I feel the individual activities of the PAWR process have quite varying levels of support *within the activity*. And I see them as fairly un-integrated across activities; typically, there is no one software application which manages and ties together this PAWR process. The one *partial* exception to this I am aware of is the SAS/PH-Clinical® product. PH-Clinical has begun to provide integrated support for the PAWR process along the lines I suggest herein (see especially Villiers, 1997). But I feel the following is still generally true in the industry today, even with PH-Clinical available (at least with the current version).

Regarding program development and execution, the SAS System is used to develop and execute the programs, but it does not provide a robust PDE to support this process. This will be further discussed in the next section of this document.

Regarding document creation, the process of populating documents with SAS outputs is not a feature of the SAS System today. Users have had to develop their own solutions which include:

- manually inserting SAS output tables into documents as pictures (which are un-editable as text);
- using custom written SAS software to engage the DDE facility to send the output to spreadsheets, which are then loaded to the documents using facilities available between the spreadsheet software and word processing software;
- using custom written SAS software to generate commands in the word processing software's language which are then executed to load the SAS outputs to the word processing document as tables (which are now editable as text).

The Output Delivery System planned in version 7 of the SAS System will take a major step forward by rendering its outputs in formats compatible with document environments. This will truly be a major, long awaited advance. But, as I understand it, even then it will be incumbent upon the user to add features to fully automate this SAS-to-document process as part of an integrated application.

Regarding the document review process, some work has already been done in this industry to integrate document management systems into the review of NDA documents, both for in-house review and particularly for external review at a government regulatory agency. Specifically, such systems now make documents come alive during review through use of hyperlinks from text to text, search functions, dynamic tables of contents, automated annotation, etc. And these systems support creating and sharing annotations in the document by multiple parties during review. Further, we are now also seeing document management systems provide web-enabled and web-exploited documents to enhance the review process. So I would not describe the document management and review activities of the PAWR process as unsupported, from the software application point of view.

But I would still call them un-integrated activities of the PAWR process. Specifically, much less thinking has occurred to date about how an automated analysis document might also be linked to the data and programs behind the analysis during the review. If this were done, then if a reviewer had questions about a particular table in the document, he/she could drill down into a *facilitated environment* to view the exact data involved in the table (i.e., the patient group involved), to view other data for the same patient group, and even have the ability to re-run the table with the data for other sets of patients or with the same patient data but with other specifications for the table (e.g., different cut points between categories of an item being reported on in the table). And a SAS PDE ---- whose initial purpose is to help SAS programmers create, access, manage, and execute SAS programs, outputs, and data ---- could also be used as the *facilitated environment* where document reviewers drill

down to, in order to gain access to the data and programs to answer their own questions about the analysis during review of an automated document.

Finally, a comment regarding electronic document submissions. If a company develops a document management application for internal assembly, management, and review of a document *during the PAWR process*, there is potential to use the same application as the CANDA tool for external review at a regulatory agency. If done this way, it is an excellent integration of software application support activities within the PAWR process. But I think the more typical situation today is that a document management system is not used in the in-house PAWR process. Rather, when document management systems are used, they tend to be part of a document publication process that occurs separately, after the PAWR process. And this way, there is no integration of document publication with internal document writing and review, from a system's perspective.

Similarly, when a Statistical CANDA is produced, I believe it is more typically developed as a separate software application following the PAWR process, and not integrated with it. I believe that a significant opportunity for time and cost savings is being lost by approaching it this way. Specifically, a robust SAS PDE would be a facilitated environment for retrieving and executing SAS programs and data, and for retrieving and reviewing the outputs of those programs and data. And this is largely what a Statistical CANDA environment is, too. So, if a company developed its NDA by having its SAS programmers operate within a robust SAS PDE, they would also be building their Statistical CANDA at the same time. This would save a considerable amount of time that is now spent building the Statistical CANDA after the fact. And it would represent a strong integration, from the system's perspective, of the program development activity with another activity of the PAWR process --- the Statistical CANDA.

So, a SAS PDE can serve multiple functions within integrated system support for the PAWR process of the pharmaceutical industry. We might even say that by integrating the PDE fully into this process, we are moving toward the concept of an application called an analysis development environment. That is, an environment which facilitates all aspects of the analysis process, not just the creation of the raw analysis tables and graphics. And the PDE portion of this full analysis development environment would have these roles:

1. Make the Base SAS/Macro Language program development process more efficient.
2. Facilitate the automated process of populating analysis documents with SAS table and graph outputs during document writing.
3. Provide the facilitated environment where reviewers arrive, when drilling down through the automated document during review, to examine the data and re-run the programs that were used to produce the analysis tables and graphs described in the document.
4. Provide a Statistical CANDA software application.

PROGRAM DEVELOPMENT ENVIRONMENT

The PAWR process is a large one. I have only been able to describe it at the top-level, and make some top-level suggestions for how it can be integrated from a system's point of view, within the space constraints of this paper. But I am able, herein, to discuss one activity within the PAWR process, specifically SAS program development, in somewhat more detail.

Much program development in the software industry today occurs within a PDE. A PDE, simply put, is a software application that provides an environment that facilitates the development of software. PDE's are provided today for many program languages and for many application development tools --- especially for object-oriented GUI tools. Program languages using PDE's today include C, C++, and Smalltalk. Development tools using PDE's today include Visual Basic, Power Builder, and Oracle Developer/2000.

The SAS Display Manager environment is a PDE for the Base SAS/Macro Language, albeit it is less robust than the better PDE's seen today. In this section, I will present ideas about what features would be valuable in a more robust SAS PDE for the Base SAS/Macro Language. My ultimate context for this PDE is its use within the pharmaceutical industry's PAWR process. But most of what pharmaceutical SAS programmers do during program development is quite generic, so these ideas basically fit a general PDE for the SAS System. Also, I note that a SAS PDE is well worth pursuing, even if it is never integrated with software support for the other activities of the PAWR process. It will shorten cycle time for the program development activity, regardless.

The SAS Institute has already developed some PDE-type support for application development in the AF/SCL environment. It is called Source Control Manager and it is experimental in release 6.12. But

there is no similar support provided yet for program development in the Base SAS/Macro Language environment, where most pharmaceutical SAS programming occurs. I am aware today that the SAS Institute is planning to provide some sort of PDE for Base SAS programming as part of version 7 of the SAS System. But I still want to include this section in my paper to foster a dialog that can lead to a full expression of desired features by the user community. Also, I understand that the Institute's PDE will not pay special attention to Macro Language programming, which is a central part of the program development process conducted by pharmaceutical SAS programmers. I will discuss additional PDE support for macro programming herein.

Common PDE Components

In general, the thinking behind PDE's has been to identify and provide automated support for those programming activities that are done repeatedly, that tend to be tedious, and that can feasibly be supported by a software application. Here is my list of such programming activities in the SAS environment:

- Write code
- Store code (especially including generations of same program)
- Retrieve code (by generation)
- Execute code to test for syntax errors
- Execute code with data, to test for logic errors
- Debug code with logic errors
- Compare generations of code for differences
- Share code with co-developers
- Save outputs, including logs, procedure outputs (Output window), and custom generated tables and graphs sent to external files --- all by generation
- Retrieve these same outputs for viewing
- Document code
- Validate code
- Orient others to the development environment

Given this list of activities, I see the SAS PDE centered around these chief components:

1. A Custom Editor
2. A Library Manager for Source Code and Outputs

The Custom Editor would have these features:

- Syntax sensitive editing - It would examine all code from the syntax perspective, and correct it, before the code was even executed.
- Color coding - It would display the code in colors, according to rules, to better reveal the structure, and thereby add clarity and meaning to the program.
- Automatic indentation - It would automatically indent code, according to rules, as another means of revealing the structure.
- Documentation window - It would provide a separate place (e.g., a pop-up window) where the programmer could write notes that describe the program. This information would be saved along with the code, and would always be available in this editor.
- Change history window - This feature would be similar to the documentation window, but it would contain notes about the changes within and across generations of the program.
- Debug facility - It would allow you to submit code for execution from the editor, and put traces on the values of variables, and then see the code scroll up line by line as it executes, and display trace variable values on-demand.

The Library Manager for Source Code and Outputs would have these features:

- Flexibility in defining libraries, such as on a per project basis. Ability to define private libraries and shared libraries, with a facility to promote and demote entries between the two types of libraries.
- Storage of all SAS programs as entries in a SAS catalog, with regular SAS catalog support for saving, retrieving, deleting, browsing, etc.
- Option to store code as the "next generation" of a given program, or as a new program.
- Similar catalog storage for all outputs when the program was executed, linked with the associated source code item. Outputs include:
    - Program logs
    - Statistical procedure output (text and graphic)
    - Externals files (text and graphic)
- Facilitated searching of log entries for notes, warnings, errors, etc.
- Storage of the source code "Documentation Window" contents, as described just above.
- A checkin/checkout facility allowing multiple programmers to obtain and work on a program in a shared library, without overwriting the other programmers' work.
- A facility to compare and report differences on two source code entries in the library (e.g., two generations for a program).
- Storage of the source code "Change History Window" contents, as described above. This feature would be related to another feature of the library which would record for each entry in the library the time and date of last change, and the identity of the last user to make the change.

- Output entries would be read-only by default. When output items were editable, then checkin/checkout support and change history support would be provided, as just above.

Additional SAS PDE Components for Macros

Many of the PDE components above might be considered standard or common support for program development, even beyond just SAS program development. But SAS programmers, certainly including pharmaceutical SAS programers, write many programs in the SAS Macro Language instead of just the Base SAS Language. As such, I envision other components in the SAS PDE to support development of macro programs. These would be in addition to having all the above common components:

- The Custom Editor would provide for a macro program (in one or more pop-up windows):
    - a list of all its macro parameters
    - the default value of each parameter, if any
    - the current value of each parameter, if any
    - a text description of each parameter, as supplied by the programmer
    - GUI type support (point and click, drag and drop) for setting the current value of each parameter
- The Library Manager would handle all this macro parameter information, along with the macro source code, as one or more catalog entries, with all support for save, retrieve, delete, browse, etc.
- The Library Manager would save the full set of parameter values as an additional output item when macro programs were executed.
- The Library Manager would have a macro documentation facility that would, for any given macro: (1) report what other macros are called from within this macro, and whether or not they are also in the same library or other available libraries; (2) report what other macros in the same library or other available libraries call this macro.

Program Validation

There are major requirements in the pharmaceutical industry for validation of programs used in data analysis. The effort put into this program validation is substantial. A SAS PDE would make a major contribution both toward insuring that validation was done, and toward getting it done more easily. This is because part of the program validation process includes providing well defined, well organized handling of the storage and retrieval of programs, and providing tight control over access to and editing of programs, including an audit trail on change history. These features would be inherent parts of a SAS PDE, and as such, any organization developing its SAS programs within this environment would be automatically addressing these aspects of program validation.

REFERENCES

Villiers, P. (1997), "New Architecture for Linkage of SAS/PH-Clinical® Software with Electronic Document Management Systems", SAS Institute, Cary, NC (in the Proceedings of the PharmaSUG '97 Conference).

Barry R. Cohen, Planning Data Systems, Inc.
PO Box 666, Ardmore, PA 19003
610-649-8701

Mr. Cohen is an independent information systems consultant, specializing in application development and other support for analytic processing. He has been using SAS software since 1980 in a variety of industries, including a focus on the pharmaceutical industry.