Paper 29

# The Pharmaceutical Program-Analyze-Write-Review Process and a SAS® Program Development Environment to Support It

**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 program outputs. This Program-Analyze-Write-Review process is a major, time-consuming process. I address automated support for it in this paper. A top-level discussion of the full process is followed by focus on support for the SAS program development component within it. This focus includes support for extracting clinical database tables into SAS data sets.


## INTRODUCTION

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

| 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 Activities Regarding Data & 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 (both in-house and by vendors) to handle clinical data collection, preparation, and management.
- Document Management Systems are used to publish New Drug Application (NDA) documents.
- Electronic Submissions, formerly called Computer Assisted New Drug Applications (CANDA), present 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 given little support when developing programs. No environment, beyond SAS Display Manager, supports them as they code, test, debug, store/retrieve, validate, and document the generations of their programs.
- Statisticians are given little support as they analyze data and write the statistical portion of NDA documents. No environment supports them as they 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 documents they are writing.
- Statisticians and other reviewers are given little support as they review the document. No environment is available that facilitates access to the document, or annotation and distribution of review comments in the document, or movement from the document text and tables back to the program and data environment to answer questions about the exact programs and/or data involved.

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. This is the subject of my paper. I will first talk about the PAWR process as a process, describing it activities and where I feel system

support is possible. This is a necessary precursor to building an effective software application to integrate and support the activities of the process. I will then present some thoughts about what a software application to support the process might look like. And here I will build upon the thoughts I included in my SUGI23 paper on this same subject (Cohen, 1998). I hope to foster more dialogs within the SAS community on this subject through this paper, just as I tried to do with my SUGI23 paper.

**PROGRAM-ANALYZE-WRITE-REVIEW PROCESS**

The activities of the PAWR process 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, debugging, validating, and documenting 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.

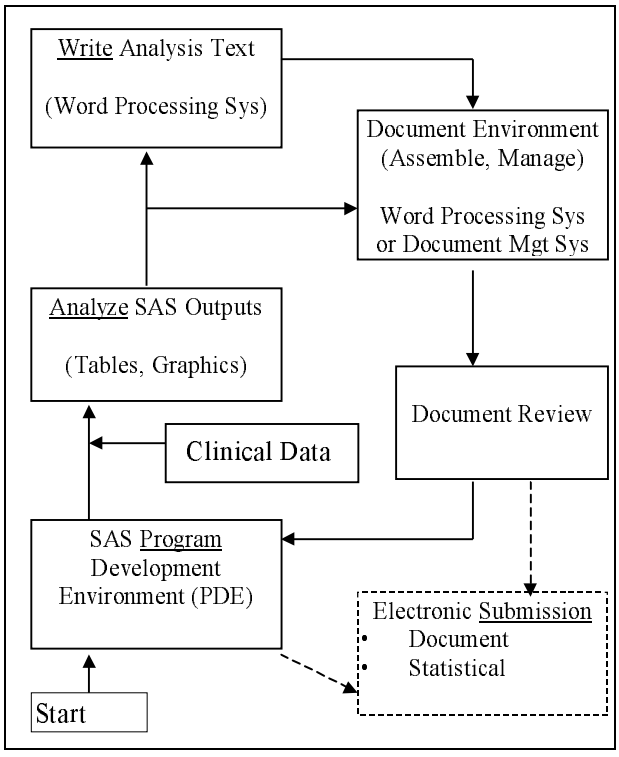


Figure 1: Program-Analyze-Write-Review Process

The process continues with the analyst writing the text that discusses the results of the analysis, referring to the tables and graphics. 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 graphics are next assembled and managed. If the word processing environment is used for this function, the tables and graphics, 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 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 author from the reviewers. Often this involves 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. The PAWR process then iterates as many times as necessary until all involved agree that the analysis is done correctly and the results are expressed in a document correctly.

Finally, once the analysis is complete and the document is written 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 Document Electronic Submission. And the statistical analysis (programs and data) may also be submitted in a Statistical Electronic Submission. I show the submission activity in a dotted line because some (many?) companies do not treat it as part of the PAWR process, but more as a separate, follow-on activity.

PAWR Activities Typically Not Integrated

I feel that across the pharmaceutical industry today, there are varying levels of system support *within each activity of the PAWR process.* And I see this support as mostly not integrated *across the activities.* Typically, there is no one software application that manages and ties together this PAWR process. In fact, I am only aware of one software application available today that has begun to provide integrated support for the PAWR process along the lines I suggest herein. That product is SAS/PH-Clinical® (Villiers, 1997). But more can be done, even by SAS in its PH-Clinical

product, and I feel the following assessment is still generally true in the industry today.

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 later in this document.

Document Writing: SAS outputs (tables and graphics) are only a means to an end for an analyst involved in the PAWR process. These outputs must become part of the document being written. But the ability to generate SAS outputs in a form that is compatible with a document environment is only beginning as part of the SAS System, with the Output Delivery System (ODS) in version 7. (Renderings in HTML format will be first, not yet renderings in Microsoft Word® format). And the ODS will be a tool for rendering SAS outputs in various document formats, but it will not be an application for tying together the process of executing a program to generate an output, and the process of positing one or more outputs in a specified document. It will still be incumbent upon the user to create such an application that fully automates this SAS-to-document process as part of an integrated application for the PAWR process.

Document Review Process: Some work has already been done in this industry to integrate document management systems into the review of documents, both for in-house review and particularly for external review at a government regulatory agency. Such systems now make documents come alive during review through use of text hyperlinks, search functions, dynamic tables of contents, support for review annotation, etc. Further, such systems are now also using the web to further enhance the review process, especially when multiple people need access to the same document. So the document management/review activities of the PAWR process are now being supported substantially through new systems.

But the support for this activity is not yet integrated with the other activities of the PAWR process. Specifically, less thinking has occurred to date about how an automated analysis document might be linked to the data and programs behind the analysis during the review. This would allow a reviewer who had questions about a particular table in the document to drill down into a *facilitated environment* to view the exact data involved in the table (i.e., the patient group involved). He/she could also view other data for the same patient group, and even 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. (For example, by using different cut points between categories of an item being reported on in the table). Further, 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 this *facilitated environment* where document reviewers drill down to, in order to gain access to the data and programs and answer their own questions.

Electronic Submissions: Many pharmaceutical companies do use document management systems today for final presentation of the analysis document to a regulatory agency. But many companies surprisingly do not use the same application for support in-house as they assemble, manage, and review the same document *during their PAWR process*. This would be an excellent way of integrating their software application support activities within the PAWR process.

Similarly, when a Statistical Electronic Submission has been produced, I believe it has more typically been a separate software application that followed the PAWR process, not integrated with it. Approaching it this way is losing a significant opportunity for time and cost savings. If a company developed and used a SAS PDE for their program development and execution, the PDE application would be an environment that facilitates retrieval and execution of SAS programs and data, and retrieval and review of the outputs of those programs and data. This is largely what a Statistical Electronic Submission environment is, too. So, the Statistical Electronic Submission could be spun off from the SAS PDE. This would save a considerable amount of time and labor, and it represents a strong integration of system support within the PAWR process. And even if a company wanted to submit the programs and data without a robust user front-end, the in-house SAS PDE could still be used to efficiently organize, manage, and eventually spin off the precise set of files to be sent to the regulatory agency.

So, a SAS PDE can serve multiple functions within integrated system support for the PAWR process. And as we move toward an integrated application for the full process, we speak more of an *analysis development environment*, or perhaps a *submission*

*development environment*. This is in contrast to a program development environment, which is narrower in scope. The larger application encompasses the PDE. It is a software environment that facilitates all aspects of the PAWR process, not just the programming that is done to create 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 Electronic Submission software application.

## PROGRAM DEVELOPMENT ENVIRONMENT

The PAWR process is large and within the space constraints of this paper I am only able to describe it at the top-level, and make some top-level suggestions for how it can be integrated from a system support point of view. But I can further discuss support for one activity within the PAWR process --- SAS program development. 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 present ideas about what features would be valuable in a more robust SAS PDE for the Base SAS/Macro Language. My context for this PDE is its use within the pharmaceutical industry's PAWR process. But these ideas basically fit a general PDE for the SAS System.

PDE applications generally provide automated support for the common programming activities that are done repeatedly, that tend to be tedious, and that can feasibly be supported by a software application. I feel that a SAS PDE in the pharmaceutical industry should have this common activity support, and also have another component that supports SAS macro programming activities. SAS macro programming presents its own unique challenges, and pharmaceutical SAS programmers use the SAS macro language often.

I first describe features of a SAS PDE to support common programming activities. I do not expect a pharmaceutical company to build these capabilities from scratch. Some or most of these features will either be available within the SAS System as SAS version 7 emerges, or an organization can acquire them in a third party, commercial off the shelf, generic PDE tool. I then describe features of a SAS PDE that would specifically support activities of SAS macro language program development. I hope that the SAS Institute adds these features to Display Manager, but I think it is more likely that an organization will have to develop this in-house, if they want it in the near term.

### SAS PDE Features for Common Activities

A SAS PDE that would support the common programming activities might be centered on these chief components:

1. A Custom Code 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 executed.
- Colored code display - 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 window could be designed to have required "fields" or text boxes, to help insure that complete documentation is written for each program. The documentation would be saved in

the library, associated 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 – This would be a set of features that are standard to most PDE environments. It would allow you to submit code for execution from the editor, put traces on the values of variables, 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.
- Storage of all SAS programs as items 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.
- A check-in/check-out 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 that would record the time and date of last change, and the identity of the last user to make the change, for each entry in the library.
- Storage of the source code "Documentation Window" contents, as described just above.
- Similar catalog storage and retrieval for all outputs created when a program is executed. The outputs would be linked with the associated source code item in a "run-set", and the user could also store a brief text description of the run-set. The run-set would include:
    - Run-set description
    - Program source code
    - Program log
    - Program listing or OUTPUT Window contents
    - Statistical procedure output data sets
    - Externals files (text and graphic)
- Facilitated searching of log entries for notes, warnings, errors, etc.
- Output items would be read-only by default. When output items were editable, then check-in/check-out support and change history support would be provided, as just above.

SAS PDE Features for Macros Activities

- The Custom Editor would provide handling for the macro parameters of any macro program. The macro parameter information listed below might be stored in a pop-up spreadsheet called a "macro parameter property sheet":
    - a list of all the macro parameters, one per row of the sheet
    - a text description of each parameter, as supplied by the programmer
    - default value of each parameter
    - current run-time value of each parameter
    - support for setting the current run-time value of each parameter; this could be via text entry directly in the sheet, of through some sort of GUI (point and click, drag and drop)
- The library would also hold the correct syntax for calling the macro, probably with default values for the parameters
- The Library Manager would handle all this macro parameter information, along with the macro source code, as multiple catalog items, with all support for save, retrieve, delete, browse, etc.
- The Library Manager would save the set of parameter run-time values as an additional output item in the "run-set" when macro programs were executed.
- The Library Manager would have a library search facility that would, for any given macro: (1) report what other macros are called from within this macro, and what library they are in; (2) report what other programs call this macro, and what library they are in.
- The Library Manager and Custom Editor would together provide the facility to automatically generate the syntactically correct call to a given macro and posit it into a program residing in the code editor.

The SAS PDE to Database Interface

In the pharmaceutical industry today, the clinical data input to analysis programs is often extracted from a database system and converted to SAS System data sets. The programming effort for extraction from the

database system to SAS data sets can be a labor-intensive process. The PDE can provide support for the common activities of this data extraction. This support, strictly speaking, might not be considered part of the PDE. It might more correctly be another component of the larger support application that the PDE is a part of. (I referred to this larger application above as an *Analysis Development Environment* or a *Submission Development Environment*). Either way, data extraction support is closely tied to the program development/execution support in the pharmaceutical world, so I include a few thoughts about data extraction support here.

I envision the following capabilities available to the user:

- Display a list all database instances available for a specified drug project
- Display a list of the database tables available for a selected database instance
- For all clinical database tables that are standard for the company (i.e., used repeatedly across protocols and across projects): Extract the data and create SAS data sets with pre-specified SAS variable names and labels. These names and labels will replace default variable names and labels that SAS/ACCESS would otherwise build from the database column names. The default names are often meaningless and potentially confusing.
- For unique (i.e., nonstandard) tables: Support extraction of the data with the ability to define custom SAS variable names and labels to override the defaults that SAS/ACCESS will otherwise provide.
- For all database tables being converted to SAS data sets, provide control over columns and rows being extracted. Allow the user to specify database columns for inclusion or exclusion. Allow the user to build "where clauses" that limit the rows extracted.
- Provide a data-encoding tool for use during extraction. This tool would first assist users to define numeric codes for text-based database columns that require numeric codes for analysis. It would then use these codes to create a new, derived SAS variable as the database table is extracted and converted to a SAS data set.

Program Validation and the PDE

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 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**

Cohen, B.R. (1998), "Supporting the Program-Analyze-Write-Review Process with a Development Environment for Base SAS® and the Macro Language", Proceedings of the Twenty-Third Annual SAS User Group International Conference, 23.

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
cohenbar@bellatlantic.net

*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.*