

Data Quality and the Big 'D'

Elizabeth Axelrod, Abt Associates Inc., Cambridge, MA

ABSTRACT

With the passage of time, it becomes more and more difficult to recall or recover the process, data, or results of a project. With the new Federal Data Quality Act, it is even more critical that the entire data process be repeatable. This paper presents an overview of the whys and hows of documenting an entire data process - the many steps involved in creating deliverables for government research projects. Examples of how to accomplish this are provided.

INTRODUCTION

Imagine that you completed a research project a year ago, and now your clients are challenging the results. They doubt your numbers, and they want to know how you came to your conclusions. Could you retrace your steps and reproduce your results? Could you describe what you did and why you made those decisions along the way? Could you find the data or the programs that you wrote?

The new Federal Data Quality Act has caused programmers and project managers to review their respective responsibilities in the area of "reproducibility of results," and to take a fresh look at the challenge of project documentation.

THE BIG 'D'

"Documentation" is a term that is often used loosely, and it can mean different things at different times. It is important to recognize the hierarchical nature of documentation and, to this end, it is useful to make the following distinctions among types of documentation:

1. Comments in the code.
Comments help organize and explain the logic of a program. Comments are useful to the program author and other programmers.
2. Descriptive Documentation.
These are the various documents that are developed during the process of file construction, and might include e-mails, memos and correspondence, specifications, descriptions of finished programs, user guides, etc. They are the building blocks of the overall project documentation, but are not necessarily organized in a useful format.
3. Process Documentation.
Process Documentation includes (1) and (2), above, as well as the 'recipe' used to construct the final file(s). It is the history of a project, recorded in sufficient detail to allow programmers and managers to retrace their steps, to reproduce results, and to provide "sufficient transparency about data and methods that an independent reanalysis could be undertaken" (language taken from the new Data Quality Act). This is the Big 'D,' and how it affects you, the programmer, is the subject of this paper.

Typically, the project leader is responsible for the design of Process Documentation. The programmer is responsible for ensuring that the steps he or she takes in the handling of data can be recreated in the future. This includes traditional practices in code development and data handling, as well as the avoidance of certain practices which are inherently difficult to document.

THE DATA PROCESS

For the purposes of this paper, the "Data Process" consists of all the steps necessary to go from raw or source data, to a final product, whether it's a file, a table, or a report. These steps might include:

- Traditional batch processing (running a SAS job on an input file, to create some sort of output);
- Moving data from one place to another (such as downloading data from the Internet, or uploading data from your PC to a mainframe);
- Transforming data interactively (in EXCEL spreadsheets, MS Access tables, or during an interactive SAS session).

The programmer's job is to record the Data Process.

KEY ITEMS TO BE RECORDED

At a minimum, you would need to record the following items for each step or program along the way:

- Date the job was run
- Name & location of input file(s)

- Name & location of program
- Name & location of output file(s)
- Number of observations created in the output file(s)
- Brief comment describing the task.

METHODS OF RECORD KEEPING

There are many ways you can maintain this information - some manual, some automated. There are even products which help you do this. I use a simple form, shown in Exhibit 1, which I call the *docbox*. Each column of the *docbox* corresponds to one of the minimum bits of information listed above. I record these items for each program I run, especially if the resulting output is a new file.

If you need to recreate the files, you've got all the information you need: a record of the sequence and location of the programs you ran, and the location and names of the source data. And if you want a reminder of what you did, you can just read down the right-hand column (comments), to see a descriptive summary of each step.

EXHIBIT 1

SAMPLE DOCBOX

PROJECT: <i>ThisProject</i>			TASK: <i>Pre-process the enrollment data for Wave 2, and Compare to Wave 1 data.</i>		
DATA: * = c:\projects\ThisProject\data\wave1\ ** = c:\projects\ThisProject\data\wave2\ PROGRAMS: # = c:\projects\ThisProject\pgmdir\wave2\					
DATE	INPUT	PROGRAM	OUTPUT	N obs	COMMENTS
5/23/2003	*enroll.txt	#sasify01.sas	*enroll.sas7bdat	23,247	SASify the raw enrollment data. Keep all vars. Look for duplicates on CASEID. (No duplicate CASEIDs found.)
5/23/2003	*enroll.sas7bdat	#stat01.sas	p.o.		Preliminary freqs, xtabs, means.
5/24/2003	*enroll.sas7bdat **enroll.sas7bdat	#compare.sas	*dubious.sas7bdat	13	Merge the Wave 1 data with the Wave 2 data, and look for records that are not in both. Save these records for future examination.
5/24/2003	*dubious.sas7bdat	#badguys.sas	p.o.		Print selected vars for the dubious enrollees (who aren't in both Wave 1 and 2).
5/24/2003	*enroll.sas7bdat *countyvar.txt	#addcounty	*enroll2.sas7bdat	23,247	Append county-level data onto each enrollment record. Print cases where no county match found.

The *docbox* is just one method for documenting a process. A *flowchart* is useful as a more visual representation of your data flow. Note, however, that if you do not include the actual paths and filenames, a flowchart only serves as a conceptual representation of what was done, and not a detailed document, or recipe, that could be used to redo the steps and reproduce your results.

BEAUTY IN A BOX

Using a *docbox* has several advantages. If you wait until the end of the data process to do your documentation, it will likely be a painful chore, and you will probably just not do it at all. But if you document as you go, it's not so bad, and, in fact, has some

collateral benefits. By taking a step back from coding and running your jobs, and really looking at your SAS logs, you might see an error or inconsistency you would not otherwise have caught. Or if you have to re-run the last few programs, you don't have to try to remember which programs to run, or the order to run them in -- it's all in your *docbox*, and you can follow it like a recipe.

There are certainly ways to help you automate the process of maintaining this kind of documentation, and many programmers I talk to have asked me if I have developed any macros or background processing that will grab internal comments and 'dump' them into the documentation. I prefer to do it manually, because it forces me to really examine the results of each of my programs, and to make sure that the results I'm getting are consistent with what I expected. I have developed some simple macros within WORD to simplify the data entry part of the task. The method you use is not important - as long as you do *something* to keep track of your data process.

CODE IS TRUTH!

Documentation is not truth... it's only what you *think* you did... only your code knows for sure. If you update your code, or re-run a job, you need to update your docbox, or its equivalent. Your documentation is only as accurate as your diligence in maintaining it. Although it is important and useful to develop documentation, keep in mind that if you want to know the exact details of what you did, you will need to look at your code.

ORGANIZE AND MAINTAIN THE 'STUFF'

The more complicated your project, the more 'stuff' you've probably accumulated: memos, e-mails, *docboxes*, printouts (LOGs and LISTs). Now that you've got all this documentation, it's worth thinking about how to keep track of it.

Keeping a disorganized stack of printouts on your floor is perhaps not the best approach! Printouts can be kept in folders or binders, and it is always advisable to maintain the SAS LOG and LIST together (including any JCL or system messages, if applicable). An isolated SAS LIST, without the associated LOG and file references, does not provide a complete picture of what was actually done.

One approach is to keep all printouts in binders with an accompanying *docbox*, or equivalent job summary document. An additional organizing tool is a 'project bible', containing all descriptive documentation, including pertinent e-mail correspondence, memos, and programming specifications.

BATCH VS INTERACTIVE MODE

Choices you make in the data process effect the Big 'D'. Programmers have options when it comes to running SAS jobs: we can use the display manager, and run our jobs interactively, or we can use batch mode, where we submit a job and run it in the background.

Running jobs interactively provides several advantages: You get quick turn-around and immediate results. It is nicely suited to 'playing' with your data, to try out new things, and develop, test, and debug your code. But it has some severe disadvantages, too. Even if you religiously save the logs from these sessions, they will not necessarily provide an accurate record of what was done. For example, if you change some system options during the session, your final log might not show the options that were in effect the last time the code was submitted.

Interactive mode is great for developing, 'playing', and testing, but when you are ready for a production run, use batch mode to ensure that your results can be reproduced.

FINAL THOUGHT

It is crucial that we develop accurate and complete documentation as we proceed through a project. Independent of the new regulation, it is always advisable to leave tracks on our path so that others may follow.

REFERENCES

For more information on the Data Quality Act, go to: http://www.whitehouse.gov/omb/infoereg/agency_info_quality_links.html

ACKNOWLEDGEMENTS

The author wishes to thank Steven Strang and Rick Sheppe for their editorial contributions to this paper.

CONTACT

Elizabeth Axelrod
Abt Associates Inc.
55 Wheeler Street
Cambridge, MA 02138
elizabeth_axelrod@abtassoc.com

MWSUG 15-T9

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.