

Paper 114

Documentation as you go: aka Dropping Breadcrumbs**Elizabeth Axelrod, Abt Associates Inc., Cambridge, MA****ABSTRACT**

Your project ended a year ago, and now you need to explain what you did, or rerun some of your code. Can you retrace your steps? Find your data, programs, printouts? Replicate the results? This poster presents tools and techniques that follow best practices to help us, as programmers, manage the flow of information from source data to a final product.

INTRODUCTION

As programmers, we're tempted to head straight for the keyboard, and start writing code. But with the passage of time, it becomes more and more difficult to recall or recover the process, data, or results of a project. So it's worth thinking about how we're going to keep track of all the steps we'll take – the many programs we'll write – and the overall flow of data – for a project. This paper addresses documentation, the most obvious method for helping us do this. Maintaining robust documentation helps us to ensure the reproducibility of our results, and in doing so, improves the quality of our work.

Concepts presented here are just some of the important steps we should take; others are described in an expanded paper (Axelrod, 2010). All were developed while working in a government research environment, but the concepts are widely applicable in other settings.

DOCUMENTATION: THE BIG 'D'

Many data users learn the importance of documentation when data are challenged, and the need arises to explain or even reproduce the results of an analysis. At that point, everyone knows what should have been done and typically laments the day that a few simple procedures were ignored.

Preparing careful, comprehensive documentation—even in the face of time and budget constraints-- is perhaps the single most important practice for ensuring that you can retrace your steps.

"Documentation" is a broad term that covers a hierarchical array of information sources:

- **Comments in the code.** Useful to the program author as well as other programmers, comments help organize and explain the logic of a program.
- **Descriptive Documentation.** These are 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 typically organized in a useful format.
- **Process Documentation.** Process Documentation includes all of the above, as well as the 'recipe' used to construct the final file(s), or product. It is a comprehensive history of the project, recorded in sufficient detail to allow programmers and managers to retrace their steps, and reproduce results. This is what I refer to as the Big 'D'.

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 that 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:

- Data receipt (receiving data from an external source, be it the internet, a client, or a colleague)
- 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).

It is the programmer's responsibility to record the data process, and here are recommendations for how to do this.

MAINTAIN A DATA CATALOGUE

Keep track of every file and document you receive or send. Even a simple form like this can hold the minimum bits of information needed. In this example we keep track of the source, date received, and a brief description of the files, and their destination filename and location. You might even want to keep passwords here, for encrypted data.

A	B	C	D	
Source	Description	Date Recd	Filename	Comments
CMS	Yr 1 Cases	8/9/2007	Final_payment_HQID.xls	delivered via e-mail from
CMS	Yr 2 Cases	8/9/2007	Year_2Final_Payment.xls	delivered via e-mail from
CMS	Yr 1 Demo Payments	11/14/2007	finaldemo.xls	delivered via e-mail from
CMS	Yr 1 Demo supplements	11/14/2007	p141339.xls	delivered via e-mail from This spreadsheet include Memorial Hospital that v
CMS	Yr 2 Demo Payments	11/14/2007	CMMdemocaxfy05.xls	delivered via e-mail from
		NOTE: All files are located in H:\HSRE\PREMIER_Stuff for Program		

Exhibit 1. Sample Data Catalogue

KEY ITEMS TO BE RECORDED FOR A DATA PROCESS

At a minimum, you 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 to maintain this information - some manual, some automated. I use a simple form, shown in Exhibit 2, 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.

Here is a sample DocBox:

PROJECT: <i>ACME project</i>			TASK: <i>Process the Wave I ACME Survey</i>		
LOCATION OF DATA: * = h:/ACME/data/survey/					
LOCATION OF PROGRAMS: # = h:/ACME/pgmdir/survey/					
RUN DATE	INPUT FILENAME	PROGRAM NAME	OUTPUT FILENAME	# OBS	COMMENTS
2/1/15	*ACME_Survey.txt	#surv001	*ACME_surv001	12,346	SASify the ACME wave1 survey data. Look for duplicates.
2/1/15	*ACME_surv001	#surv002	Printout	n/a	Examine the duplicates.
2/4/15	*ACME_surv001	#surv003	*ACME_surv002	12,340	Remove duplicates. (6 duplicates removed)
2/5/15	*ACME_surv002	#surv004	Printout	n/a	Run preliminary freqs and means. Check skip patterns.
2/6/15	*ACME_surv002	#surv005	*ACME_surv003	12,340	Apply recodes: Set legitimate skips to special missing values.

Exhibit 2. Sample ‘DocBox’

The DocBox is only one method for documenting a process. A flowchart is useful as a more visual representation of a data flow, and is particularly helpful if you have a complex data process. 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.

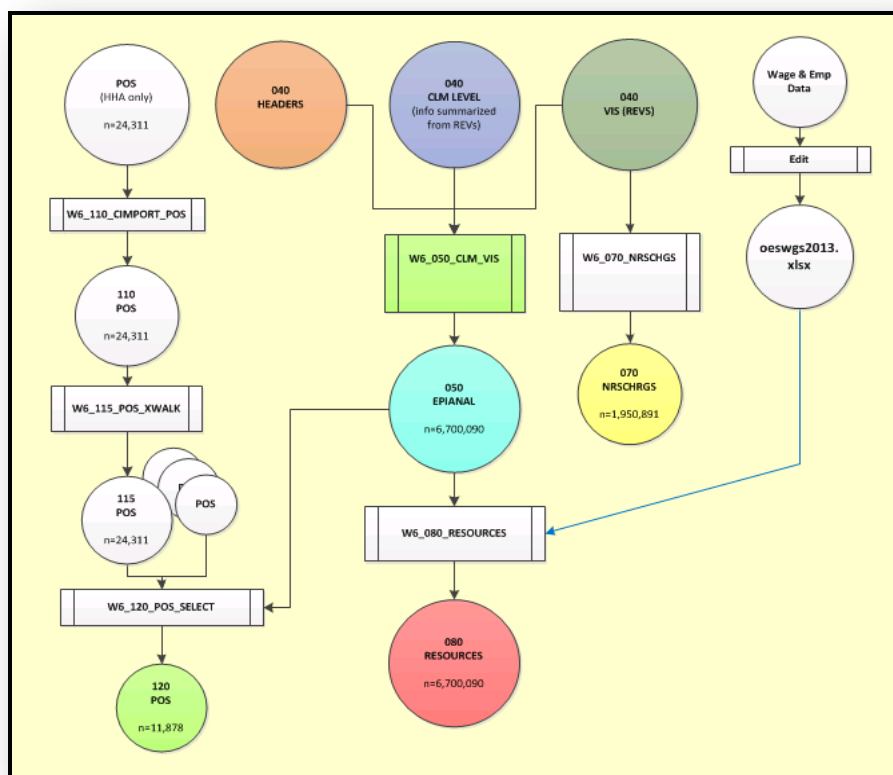


Exhibit 3: Sample Flowchart

Whatever method or methods you choose, here are some basic principles to keep in mind.

DOCUMENT AS-YOU-GO

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.

STEP AWAY FROM YOUR CODE

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 and it's only as accurate as your diligence in maintaining it. Keep in mind that if you want to know the exact details of what you did, you will need to look at your code.

CONCLUSION

Follow the simple techniques outlined here and you will leave a trail of crumbs behind so that you – and your team – will know how you got here from there, and can find your way back again!

REFERENCES

Axelrod, Elizabeth. "Boot Camp for Programmers: Stuff you need to know that's not in the manual - Best Practices to Help us Achieve Reproducibility", *Proceedings of the SAS Global Forum 2010 Conference*.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

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

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.