SOCIETY OF ACTUARIES

Article from:

CompAct

April 2009 – Issue 31

The End Users Justify the Means III: The Search for Problems

By Mary Pat Campbell

Ah yes, our friends the auditors and testers, looking for where we screwed up in our work. Our favorite end users of our spreadsheets, huh?

Fittingly, this installment in my series of articles on spreadsheet structuring and design will be the shortest, as the goal in dealing with this particular set of end users is this: To spend as little time as possible with them.

Time spent with an auditor is definitely not productive, as necessary as it may be, and the time spent with a tester is mainly a function of how much one has messed up one's spreadsheets. Productive time can be spent with testers, but ideally you want to hear little more than, "Yeah, looks good to me."

Keep in mind that the goals of auditors and testers are different. In general, auditors are looking at static spreadsheets, making sure the inputs are appropriate, spot-checking some of the interim calculations, and making sure the outputs are reasonable and end up in their proper places either in the next spreadsheet or software package in the line, or in financial reports.

Testers, on the other hand, generally are looking at the dynamic possibilities of a spreadsheet, having several different possible input sets. In addition, some testers are not just checking that the calculations look right, but also actively try to break the spreadsheet. At least, that's how I test spreadsheets.

So let's look at some general principles in satisfying these end users, and keeping them from bugging you:

1. Clear documentation for the spreadsheet, ideally within the spreadsheet itself.

The following comes from the article "Is This Spreadsheet a Tax Evader?" as a checklist for spreadsheet auditors in desirable documentation for a spreadsheet (or general calculation package):

- the application's purpose, what it does and how it does it.
- any assumptions made in its design.
- what standing data constants (e.g., tax, duty and exchange rates) are used and where they are held.
- who developed it and when.
- when and how it has been changed since being brought into use.

The author of the aforementioned article, Raymond Butler, goes on to say that "Clear instructions for use should also be present."

I have found it useful to have a single sheet within an Excel Workbook to serve as my overall documentation and version control page. There is an implementation example on page 48 of "Spreadsheet Modelling Best Practice" (see the Sources at the end of the article.) I think the example given is a bit verbose and should have each version information be a single row, with the information of version number, changes made, date of change, by whom, etc. each be in separate columns. To be sure, it is not as pretty as the IBM format in the paper, but it makes it easier in terms of organization, and one is unlikely to miss a field of information.

Part of the reason others and I advocate having the documentation within the spreadsheet itself, as opposed to a separate document, is that one knows the documentation is synchronized with the spreadsheet itself. It is too easy to be working on a spreadsheet and totally forget about a separate document until well after the fact, so that separate documentation is rarely up-to-date. There is little excuse when the documentation is within the spreadsheet itself.

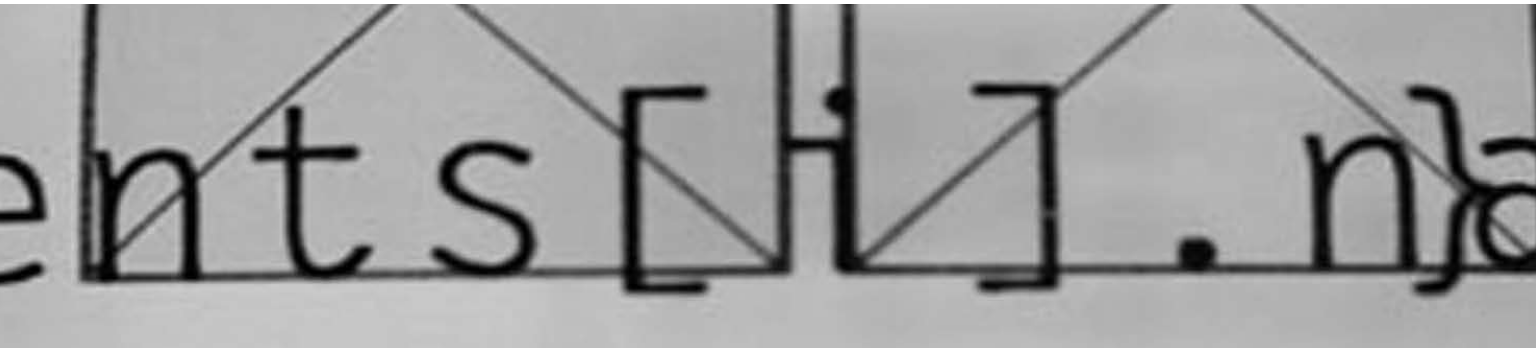
As well, if the documentation is within the spreadsheet, you should have less e-mail from the auditor or tester asking you to resend applicable documentation. Remember: the goal is to spend as little time as possible with these people.

2. Inputs should be clearly defined, all constants explicitly declared, and all input visible.

For both testers and auditors, the inputs need to be even more clearly defined than with general public users or even numerate decision-makers. There may be many

CONTINUED ON **PAGE 14**

Mary Pat Campbell, FSA, MAAA, is a vice president at The Infinite Actuary. She can be contacted at marypat.campbell@gmail.com.



"inputs" those users never change, that are effectively constants. However, both auditors and testers need to know about these numbers; auditors to check accuracy or reasonability, testers perhaps to change these items to test robustness of the spreadsheet.

Sometimes one may have constants defined within VBA macros for the spreadsheet. I recommend having an initialization macro automatically writing out the constants found within the VBA code onto a specific sheet within the workbook, making the implicit explicit.

All inputs and assumptions should be visible. It is difficult to check what one can't see. Of course, any auditor or tester worth his salt should be able to unhide everything (and yes, Excel password protection is a joke and easily crackable … but still, it doesn't look good to be trying to hide something from a tester or auditor. If you're worried they'll mess up your work, that's what backups are for. I highly recommend using those.)

As well, it would be nice from a tester's point of view if all input fields were labeled, in terms of Cell or Range Names. You can paste all named cells and ranges onto your documentation page. The benefit to the tester is that this helps automate the testing process; the tester can write their own macros to fill in these ranges with appropriate test sets of inputs. It can also make one's life easier if one has to maintain the spreadsheet, but more on that in part four of this series.

3. Outputs should be clearly defined, well-organized, and all visible … as well as reproducible.

Not much different from the inputs section, of course. However, where one may have concentrated more on graphical outputs, and a clean interface with regard to users that would use this information to make decisions or to learn about various issues, the point here is that the actual numbers need to be visible.

Testers and auditors both often have separate spreadsheets or software that independently calculates results for specific input sets. They are generally going to want to see the actual numbers to compare against their own calculations, as opposed to a graph. And when you are giving a wall of numbers to a person to interpret somehow, if you don't want to spend your time explaining and re-explaining which outputs are where, you should have them organized in a logical, clearly labeled way.

Another note here: you should not have function calls that are irreproducible. You may think this an odd exhortation, but in this age of stochastic modeling (yes, some people use Excel for this purpose) one may program in a pseudorandom number generator that one does not seed. Such as, oh, Excel's RAND function. This is bad practice from many points of view, but one of the worst is that this cannot be audited. I cannot think of another type of function call that may be variable other than something related to current time, but the point is that testers, auditors and you should be

able to get the exact same results from the spreadsheet using the same inputs. If this does not occur, redesign your spreadsheet.

There's really not much more to it than that. The main point is for one's processes to be transparent and organized. The point here is not to make the inputs and outputs easily interpretable so much as easy to find so as to make it easy to compare to what should have resulted. Nothing should be hidden, as a matter of course, though much of your work will likely go unexamined.

I did not go over how one should do a thorough audit or test of spreadsheets here. I will point to the book Spreadsheet Check and Control, which I reviewed in July 2008 in CompAct. I may revisit the concepts found in that book, as well as my own testing experience, at another time. As a spreadsheet tester, I found access to the actual VBA code, and a good programming style on the part of the creator, the most helpful in my testing endeavors … mainly because by looking at the code itself, I could see what would break it. Much more efficient than trying a whole bunch of input sets willy-nilly (though I did that for checking out calculation behavior in normal ranges.)

A final thought—if you know your spreadsheets are going to be audited, I recommend having a separate person, and if at all possible, two other people, to test your spreadsheets before the fateful day arrives. Because, of course, the best way to reduce time with auditors is to not have any mistakes in one's spreadsheets. Alas, to reduce time with the auditors, you may have to increase your time with the testers … ah, the spreadsheet author's lot is not an easy one. ■

SOURCES

Butler, Raymond J, CISA. "Is This Spreadsheet a Tax Evader?" Proceedings of the 33rd Hawaii International Conference on System Sciences - 2000 *http://www.eusprig.org/hicss33-butler-evader.pdf*

Read, Nick and Batson, Jonathan. "Spreadsheet Modelling Best Practice." April 1999. Institute of Chartered Accountants for England and Wales. *http://eusprig.org/smbp.pdf*

European Spreadsheet Risks Interest Group: page of news stories of spreadsheets with costly mistakes. *http://eusprig.org/stories.htm*

Campbell, Mary Pat. "Spreadsheet Check and Control: A Review" CompAct, July 2008, pages 13-14. *http://soa.org/library/newsletters/compact/2008/july/com-2008-iss28.pdf*