PhUSE 2015

Paper IS05

If I had a hammer

Diana Stuart, Veramed, Twickenham, UK

ABSTRACT

Now that you've learned to program in SAS, you have the tool to do your job. So what's next? Of course there are many different career paths that are open to the junior statistical programmer, but the majority look to progress from being a member of a study programming team to leading. What other skills can you develop that are useful, to allow you to do your job well, and make progress?

INTRODUCTION

On the day that I sit down to write this, I am settled with my laptop on the sunny terrace of my other half's golf club watching him wrestle with a knot in the lace of his golf shoe. You know the sort of knot, one that's been there for weeks, cemented with dried in mud, and seemingly immovable. I take it from him and try to wiggle it in an attempt to loosen the crud and ease the knot out, but before I have the opportunity to make any kind of progress, I have the stubborn article removed from my hands and it is being stabbed with the sharp end of a golf tee. The inevitable happens, and it's only when blood is leaking from both thumbs does he give up. My other half is a low handicap golfer, he knows what to do with a golf tee and how to use it to great effect. Apparently he also knows how to do damage with it, mostly to himself. The point is, in life we have many tools at our disposal, and our success is dependent on how effectively we use them. As programmers in the world of Biostatistics, SAS is our main tool, and how we use it is consequential to our success. It is the like the hammer in our toolbox, and whether we use it to build robust and elegant structures, or something that just about hangs together and will fall over at the first sign of dirty data, is down to how we develop our skills, and apply what we have learned from experience and shared knowledge. Once we have begun to master SAS, adding other tools to our toolbox assists us in the progress along our career paths.

WRECK IT RALPH OR THOMAS CHIPPENDALE?

So you've learned to code a bit, let's take stock of what you know. Are you the toddler that has just worked out how to bang the round peg into the round hole, or are you the master craftsman? Somewhere in between one would guess. The path of experience is about taking on board the mistakes of others and yourself, appreciating a good programming practice and constantly looking for new methods and efficiencies? Who hasn't looked at their old programs and thought, what was I thinking? As your experience grows, more often you will look at old programs and think, that was nicely done. Although the chances are you will think, that was nicely done, but it would be improved if I did this! The perfectionists in all of us will always want to have a say.

Whilst we are looking at how your skills are developing, it is worthwhile mentioning that familiarisation with a good programming practice and taking it's guidance into account is always a good habit to get into. Adhering to the guidance of a good programming practice isn't cause to groan and roll one's eyes. They are developed to ensure consistency and quality, to minimise mistakes and bad practices, in short they exist to help with the health and safety of your hammer. If you feel the need to question the logic behind any practices, I would encourage this. Much better to find out why something is done a certain way, than moan about it, and maybe even ignore it. The chances are there is a story to go with the logic, and any negative outcome would have been followed by the strong desire to not repeat any such outcome again. Most companies have a good programming practice, and if yours doesn't, PhUSE has developed one for all to view and contribute to.

ASK NOT WHAT IS MY LEAD PROGRAMMER DOING FOR ME, BUT WHAT CAN I DO FOR MY LEAD PROGRAMMER

Trite I know. As we start out along the path of becoming a programmer, we need to ask questions and accept guidance. Unless you become the all seeing eye of SAS programming and study management this will never change, but the types of questions and guidance should. You begin with needing guidance on when to start a task,

and how to do it. With experience, the spoon-feeding should lessen and ideally you will become more pro-active, anticipate issues and understand the dependencies of tasks.

So what is your lead programmer expecting from you? As a novice hammer welder, your lead programmer will be giving you bite-sized pieces of work where it is easy to control the amount of damage you could potentially cause from any errors. As their trust in your ability grows, along with your knowledge and experience, you and your hammer will be let loose in more risky parts of the project. They are expecting you to notice any inconsistencies and ambiguities in the specifications, and to bring them to their attention. They are expecting you to produce quality, robust, readable and where appropriate, reusable code. They are expecting you to self-check your work. This last point is an important one. At first, your lead programmer, or mentor if you have one, will be there to pick you up on any silly mistakes; it is dangerous to become reliant on this safety net. The expectation is that you will learn from your mistakes, and have an understanding of what fit for purpose means.

What can you do for your lead programmer? You can ensure their time is not spent double checking your work by necessity, understand what processes need to be followed, and by association knowing what documentation needs to be completed, and that you complete it without needing to be asked several times.

IT'S ALL ABOUT ME!

What are you doing for yourself to ensure you progress? Your career is not a box-ticking exercise. You will not be promoted because you have been in your position for a certain amount of time. It is important to consider what it is that you want for yourself.

Someone who is good at their job and continues to be good at their job in all probability enjoys their job. We can only deceive ourselves for so long, so disliking your job and being good at it rarely go hand in hand. So, which aspect of your position do you enjoy and where do your strengths lie?

Let's assume that you are happy in your career choice as a programmer, that your SAS skills are developing nicely, and you've experienced some study work. The discussion with your line manager of what next, most often will lead in one of two directions. Either you are most motivated by the technical aspects of your position, or you want to develop some project management skills and lead a study. Or why not both? The majority of programmers develop skills to lead a study or project, even those whose interests point towards the more technical side, as tool development and process improvement require project management.

Having taken note of what you can do for your lead programmer, helps you recognise the types of behaviours and skills you need to develop to become a lead programmer. A lead programmer needs to be organised, understand and follow processes, and communicate well with other functions.

A lead programmer will be looking to answer these type of questions (these are a just a few examples):

- At kick-off, is the programming team aware of the following:
    - Timelines
    - Where the documentation is stored
    - Which programming area to use
    - How to raise issues
    - Expectations of quality and self-checks
- Are other functions aware of how the programming timelines fit in with their timelines?
- Are their dependencies?
- Is there a record of issues raised and the decisions made?

DEVELOPING A PERSONAL TOOLKIT

Once you've become familiar with the idiosyncrasies of your internal programming system, the standard operating procedures (SOPs) you are expected to follow and the documentation you are expected to complete, you can add a few more tools to your toolbox.

I once read that organised people always have Ziplock bags. As a user of Ziplock bags I would agree, however, a Ziplock bag isn't going to help you deliver your deliverable to your customer. Your company may have some standard tools available, however, you may need to develop your own so that you can take ownership of your ability to deliver quality and be organised. Here are some examples:

Task specific check lists

Self-check quality check lists.

Decision logs; keep a record of the questions you've asked and the decisions that were made.

Modules of code, for example a log checker.

Spreadsheets that track your personal work load. Being able to flag as early as possible that your workload is unrealistic, or that a particular task has taken longer than expected will be appreciated by your lead programmer.

Timelines – you don't need to have a personal MS project set up, but using your calendar to keep note of when tasks are due is a good idea.

Develop your own filing system for email, and use it.

The SOPs that you are following often give you ideas for your personal toolkit.

CONCLUSION

You can program in SAS. Recognising that how you use that skill, the tool to do your job, is important. As you progress along your career path, in all likelihood, towards leading a study or project, adding skills, tools to your toolbox, is a worthwhile exercise.

CONTACT INFORMATION

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

Diana Stuart
Veramed
5th Floor Regal House
70 London Road
Twickenham TW1 3QS
Work Phone: +44 (0)20 3696 7240
Email: diana.stuart@veramed.co.uk
Web: www.veramed.co.uk