**PhUSE 2012**

**Paper IS01**

# The Logic of a Programmer – a Case Study

Julie Oates, Quanticate, Hitchin, UK

## ABSTRACT

As a programmer we are requested to become involved with many programming activities on a daily basis centered a round dataset or display generation and quality control (QC). Having the opportunity to develop a process/system which can be used by others is rare. To build any system a lot of in depth thought is required before any programming begins. We take the requirements and build a robust system to address each potential scenario that may arise, including some which should not! I will use the case study requirements in my presentation to help me describe this in further detail.

## INTRODUCTION

Within this paper I will use a case study to explain how I as a programmer work through a simple request to build a robust system using logic, SAS, UNIX and experience. It won't include all aspects of the system but it will provide a flavour of how this was built and other considerations.

## CONTENTS

- Requirements
- The "thinking programmer"
- Breaking the task down
- Example UNIX/SAS code
- Documentation
- Audit file
- Benefits of the process

## REQUIREMENTS

The client provided a simple top level request about the new system they required.

> "To build a robust system which reacts to display creation; finds and runs the related QC programs automatically and compiles the information in one central place. This needs to work for multiple studies with differing standards."

It is my role to consider all scenarios and tease out all the nuances of the system to create a system (the batch program).

## THE THINKING PROGRAMMER

How do I do this? I take the request and brainstorm all possible questions/issues and other considerations. I find this approach really helps me focus on the requirements more clearly and able to identify areas for the client to consider. Over the years I have come across programmers who need to be spoon fed the information and are not able to think through what has been requested of them. I think this is a fundamental part of our role and we should all be encouraged to engage our brains before starting a request.

Some of the questions that jump into my mind when considering such a request are:

- What aspects do I need to consider when creating a robust system?
- Do I need to restrict who can run it?
- How will I monitor who runs it?
- How will this work across the different reports of data?
- How am I going to gather the required information?
- How will I link the information?
- What will be the default settings?

- Where will I define the default settings?
- What is the new directory structure?
- What sub folders are required?
- What will be the QC naming convention?
- Do I need to consider switching code on/off?
- How can I maintain and update the QC information?
- What are the assumptions?
- Are there any limitations to the system?

## BREAKING DOWN THE TASK

After all the brainstorming it's time to gather all the questions and create an order to these, i.e. in terms of how the batch program will work. The client should be kept informed of progress and any issues discussed in detail to ensure the initial top level requirements have been interpreted correctly. You don't want to waste time creating an amazing system if the client didn't require this in the first place.

I've broken down the items as follows and we shall consider each item separately:

- Setting up the environment
- Directory structure
- QC programming naming convention
- The main components of the process
  1. Setting up the QC reporting environment
  2. Checking the displays vs QC programs available
  3. Copying QC programs from MASTER directory
  4. Running the QC programs
  5. Checking the QC logs
  6. Copying the information back to file

## SETTING UP THE ENVIRONMENT

To begin with we need to set up the environment, but to do this we need to first consider how and where we are going to define this information to enable the environment to be set up with ease.

Having one key program defined (GLOBPROT.SAS) with all the relevant information means that we can use this key program to set up the environment but also utilize this program in every other program thereafter. This means that we can have a suite of standard programs which call the specific GLOBPROT.SAS which has all the settings for that specific reporting delivery.

What information do we require to be included in this specific GLOBPROT.SAS for our client?

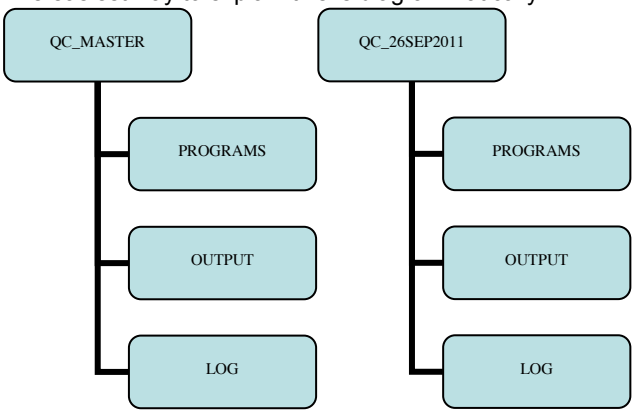
- Date of the reporting effort: e.g. 26SEP2011
- Type of deliverable: e.g. SRT, CSR etc
- Standard "Client" macro library path
- Formats
- Subsetting code
- Macro vars including:
  - outdir_lib: Display directory
  - m_lib: QC Master directory
  - q_lib: QC Reporting effort directory
- Anything else relevant to our specific reporting needs

## DIRECTORY STRUCTURE

When developing a directory structure keep it simple, you want the users to become familiar with this new directory structure immediately.
There will be a directory created called QC_MASTER and underneath this subdirectories PROGRAMS, OUTPUT, LOG etc
Note: Underneath the QC_MASTER/programs the master copies of the programs will reside.

For each reporting delivery there will be a directory called QC_*snapdate* (e.g. QC_26SEP2011) and mirroring the QC_MASTER directory the same subdirectories will be created.

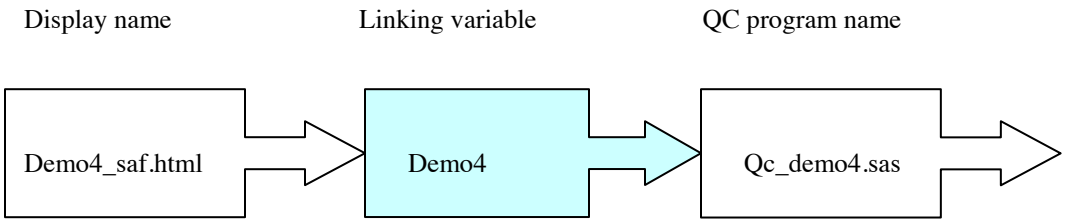
The easiest way to explain this is diagrammatically:




## QC PROGRAMMING NAMING CONVENTION

Again there needs to be a simple method used so that it may be applied to all with ease.

For this particular client the display name followed a specific pattern so it was easy to apply.
By taking the original display name to be QCed and stripping this back to create a linking variable, the QC program name then can be assigned.




The linking variable is a key variable within the program. Without this the batch program will not function properly. This variable will enable us to define the status flag for each display within the batch program, this will be discussed later.

## THE MAIN COMPONENTS OF THE PROCESS

There are six main components of the process:

1. Setting up the QC reporting environment
2. Checking the displays vs QC programs available
3. Copying QC programs from MASTER directory
4. Running the QC programs
5. Checking the QC logs
6. Copying the information back to file

The batch program was developed with the above in mind but by allowing each of the components to be driven independently using macro variables enables as much flexibility with the code as possible. The macro variables are highlighted in bold below:

- **P1_NEWDIRYN=Y:** Setting up the QC reporting environment
- **P2_QCVSOUTYN=Y:** Checking the displays vs QC programs available
  - 2a] Displays to be QCed
  - 2b] QC programs already in QC reporting effort
  - 2c] QC logs already in QC reporting effort
  - 2d] QC programs in the MASTER area
  - 2e] Incorporating the QC spreadsheet
  - 2f] Creating the match and defining the status
- **P3_COPYQCYN=Y:** Copying QC programs from QC_MASTER/programs
- **P4_RUNQCYN=Y:** Running the QC programs

- **P5_CHKLOGYN=Y** Checking the LOGs

Note when P2-P5 are run the QC spreadsheet is updated after each of these tasks has been performed. This ensures the QC spreadsheet always reflects the latest information gathered.

For the remainder of the paper I will concentrate on P1-P2 and as a whole discuss the rest when identifying the status of each display.

**SETTING UP THE QC REPORTING ENVIRONMENT**

This is driven by macro variable P1_NEWDIRYN.

At this point in the paper I will be introducing examples of the code from the batch program, specifically the UNIX commands. These need to be surrounded by the SAS code in bold below to allow the Unix commands to be run as part of the SAS batch program.

The code below creates the directories (mkdir) then copies (cp -p) the globprot.sas from the MASTER area to the reporting effort retaining the file permissions.

```
%sysexec %str(
                cd &m_lib;                              QC_MASTER/programs
                cd ../../;
                mkdir QC_&snapdt2;                      QC_26SEP2011
                cd ../QC_&snapdt2;
                mkdir programs;                         QC_26SEP2011/programs
                mkdir

                cp -p &m_lib.globprot.sas &q_lib.globprot.sas;
                );
```

The code below copies the qc_plan.csv template file from the MASTER area to the reporting effort ONLY if it doesn't already exist in the reporting effort area. It also renames it to include the reporting effort snap date.

```
%if %sysfunc(fileexist("&q_lib.qc_plan_&snapdt2d..csv")) eq 0
%then %do;
    %sysexec %str(
                    cp -p  &m_lib.qc_plan.csv
                    &q_lib.qc_plan_&snapdt2d..csv;
                );
 %end;
```

**IDENTIFYING THE INFORMATION FOR THE DISPLAYS TO BE QC'ED (2A)**

The code below navigates to the display output directory (cd) , then opens up each display and extracts the date/time stamp when the file was run (perl). The reason we didn't take the Unix date/time stamp was because the displays could have been copied to the display folder which wouldn't accurately reflect the display information.

```
%sysexec %str(
                cd &outdir_lib;                         Display directory
                                                        Internal date/time stamp
                perl -nle 'print "$ARGV: $&" if /(0[1-9]|[12][0-9]|3[01])
                (JAN|FEB|MAR|APR|MAY|JUN|JUL|AUG|SEP|OCT|NOV|DEC)
                (11|12)\ ([0-1][0-9]|2[0-3]):([0-5][0-9])/' *.* > &q_lib.dd.txt;
                );
```

Note: The perl code was provided by Tomas Demcenko which saved me writing additional SAS code.

**IDENTIFYING THE QC INFORMATION (2B-2D)**

The code below navigates to the reporting effort directory, creates a list of the QC programs only (ls -1) and a long list of the QC logs (including date/times) in this area (ls –el). Similarly navigate to the MASTER directory and creates a list of QC programs available.

```
%sysexec %str(
                    cd &q_lib ;                          QC snap directory
                     ls -1 qc_*.sas > &q_lib.qcprog.txt; 2b] QC programs
                     ls -el qc_*.log > &q_lib.qclog.txt; 2c] QC log

                      cd &m_lib;                         QC MASTER directory
                     ls -1 qc_*.sas > &q_lib.qcmaster.txt; 2d] QC programs
                    );
```

**QC_PLAN INFORMATION (2E)**

The data from the QC plan spreadsheet is imported into the batch program to maintain existing information. The following are a list of some of the variables within the spreadsheet with example entries.

Automated:

- DISPLAY: demo4_saf.htm
- DISPLAYINFO: /project/study/output/26sep11_srt/
- LINKVAR: demo4
- DISPDT: 28SEP11:16:20:00
- QCPROG: qc_demo4.sas
- QCLOG: qc_demo4.log
- QCDT: 01OCT11:04:09:00
- STATUS: 0
- STATUSD: QCed
- ORDER: 28
- AUTO_COMMENTS: 1 MESSAGES - CHECK LOG (0 ERROR 1 WARNING 0 UNINIT 0 REPEATS)

Manual:

- MANUAL_COMMENTS: All values matched with the actual output
- QCPASSED_YN: Y
- QCAPPROVER: JO
- QCAPP_DATE: 01OCT11

**COMPILING THE INFORMATION**

Here is a summary of all the files created so far which will be used in the batch program to identity the status of each displays

- dd.txt: Displays to QC (*):
- qcmaster.txt: QC programs in Master area
- qcprog.txt: QC programs in reporting effort area
- qclog.txt: QC log (*) in reporting effort area
- qc_plan*.csv: Existing QC_PLAN (*) contents

(*) The date/time derived variables are used to also identify the status and to ensure the QC information occurs AFTER the display has been created for example.

**CREATING THE MATCH AND DEFINING THE STATUS (2F)**

The table below provides an example for each status setting:

| DISPLAYS TO QC | QC PROGRAMS AREAS | | |
|---|---|---|---|
| 19APR12 (SRT) | QC_MASTER/ programs | QC_19APR12/ programs | **Status** |
| demo4_saf.html | | | 4 = QC program required |
| demo4_saf.html | qc_demo4.sas | | 3 = Copy from QC_MASTER |
| demo4_saf.html | qc_demo4.sas | qc_demo4.sas (no log file) | 2 = Run QC program |
| demo4_saf.html (20APR12) | qc_demo4.sas | qc_demo4.sas (log file date 21APR12) | 1= QC run, Log/output need checking |
| demo4_saf.html (20APR12) | qc_demo4.sas | qc_demo4.sas (log file date 21APR12) | 0 = QCed (based on manual cols) |

## THE IMPORTANCE OF DOCUMENTATION

Although the initial request was simple this did develop into a robust system which could be used by all. To ensure the system can be easily used by all users this system needs to be clearly documented. Any assumptions need to be explained and specific limitations of the system need to identified. Examples about how to use the system will help the user work with the batch program. This should be a living document which is updated alongside any system modifications.

## AUDIT FILE

In addition to the standard requirements I created an audit file which collates information every time the batch program is run. This way I could monitor the use of the program and the type of information which was required using the following 5 categories: AUDIT DATE, AUDIT TIME, USER, TASK and ADDITIONAL INFORMATION

Examples of the contents of TASK are as follows:

- P1_NEWDIRYN=Y: Created directory QC_22NOV11 and subdir programs, output and log. Copied: QCSETUP.SAS, GLOBPROT.SAS and QC_PLAN.CSV from QC_MASTER/programs
- P2_QCVSOUTYN=Y: Identify which displays have QC programs in place, which need copying over from QC_MASTER/programs and which need developing
- P3_COPYQCYN=Y: Copied over QC programs from QC/MASTER/programs
- P4_RUNQCYN=Y: Ran QC programs
- P5_CHKLOGYN=Y: Checked logs

Additional information for P3 and P4 are the QC program names

## CONCLUSION

The benefits of the new robust system are:

- Consistency between each delivery is crucial, this enables easy navigation between the snaps and programs
  - QC directory structure
  - QC programming convention
  - Display naming convention. By creating a system dependent on a naming convention, if the client deviates the process doesn't work. This was intentional to encourage the client to maintain a consistent naming convention.
- Ease of running in a separate reporting effort
  - Fundamentally it is the same QC programs just using a different globprot.sas. This allows reuse of QC programs with increased confidence
  - Having a contained area for each reporting effort means this can easily be referenced at any time.
- One central area (MASTER) with all QC programs

- This is mainly an automated system with the added benefit of allowing certain elements to be manually overridden. The only exception to this is the QC spreadsheet which only allowed 4 manual fields
- Having an automated driven system can be hugely time saving, for example setting up the area and running the QC programs can take minutes not hours. Within minutes of SRT displays been delivered for QC, I can identify displays without QC programs and have run and collated the findings of those with QC programs and collated all these findings to update the QC plan.
- The system had many automated benefits in addition to the initial requirements

## CONTACT INFORMATION

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


| | |
|---|---|
| Author Name | Julie Oates |
| Company | Quanticate |
| Email: | Julie.Oates@quanticate.com |