Paper CT04

Don't cha wish your tracker was clean like mine?

Anisa Jeetoo, Veramed, Twickenham, UK

ABSTRACT

In order to produce reliable and high-quality datasets and TFLs for submissions, multiple stages of quality control are necessary. Self-checking, independent QC, such as double programming, and a high-level review are all important parts of this process. As required by regulators, QC progress needs to be recorded. This usually happens in a QC tracking spreadsheet. But can information that is manually entered into this auditable document always be trusted? Errors can occur, from typos to missing information or run dates recorded in the wrong order. We need a way to QC the QC tracker! At my company we created a tool that performs various checks to ensure that our QC tracker has been completed correctly. This user-friendly utility is executed via the command line which calls a SAS program to check for issues. Findings are then presented in a spreadsheet so that any discrepancies can be easily resolved.

INTRODUCTION

As statistical programmers we are responsible for producing datasets and tables, figures and listings (TFLs) which are ultimately used by authorities to make important regulatory decisions. These deliverables must be accurate and in order to achieve this we must perform rigorous quality control (QC). QC checks will often involve self-checking, independent double programming and high-level review. These important steps may lead to adjustments to our programs so that they are corrected before being seen by a regulator.

To remain transparent and track the progress of all deliverables, any comments leading to these adjustments must be correctly documented. Names of the production and QC programs, the dates that the comments were made and the names or initials of the programmers who made the comments are recorded to make the QC process efficient and robust. There are several ways that companies choose to store this information; one way – which this paper will focus on – is the use of a Microsoft® Excel spreadsheet as a QC tracker: a document we use to make sure that our datasets and TFLs are programmed accurately.

While a study is ongoing, the QC tracker plays a significant role in assigning production and QC tasks, retaining QC comment history and can be useful to view all datasets and outputs to be produced in one central place. But what if there are inaccuracies in this document? Manually checking this document is possible, but it is difficult to do and prone to error: so why not create a tool to do this job for us?

Veramed is a company that works and collaborates with different companies many of which choose to record their QC activities in an Excel spreadsheet, hence we have developed a SAS® macro tool which performs various checks on the QC tracker to ensure that the information which we enter is free from errors. This tool has been designed to run in a Linux environment within a Bash shell wrapper which allows users to run the tool via the command line without having to manually update the code, making the code more robust and the tool quick and easy to run. The tool produces a Microsoft Excel document detailing the errors which were found.

We will now investigate two types of errors that can occur in a QC tracking document: human errors and QC process errors.

When referring to datasets, this paper will focus on ADaM datasets following CDISC standards, however similar principles can be applied to other sponsor specific standards as required.

COMMON QC TRACKER ERRORS

HUMAN ERRORS

As with many documents, human error in a QC tracker can be common and is often difficult to detect when manually reviewing. Examples include typos in program names and incorrect run dates being entered, which can happen when programs are re-run, but the update is not reflected in the QC tracker, as entering is a manual process.

QC PROCESS ERRORS

In some cases, the QC tracker is completed correctly but can highlight issues in the QC process itself. Examples include: QC programs failing to have been re-run after the production program has been updated and re-run, or when outputs have not been re-run after the ADaM datasets which are used to create them have been re-run. Another serious error occurs when an ADaM dataset has been run before ADSL – meaning that any ADSL variables included in the dataset might be out of date. These types of errors can take longer to resolve.

THE QC TRACKER CHECKER TOOL

We created a tool which assists us in flagging any human or QC process errors in the QC tracker (see Figure 2 for an example QC tracker). The resulting output from this tool is a Microsoft Excel spreadsheet which highlights these errors.

To make this tool user-friendly to run, the SAS program is wrapped in a Bash shell script which can be run by anyone, using the Linux command line. To run the tool, all users have to do is:

1. Save the appropriate sheets (for datasets and TFLs) in the Microsoft Excel QC tracker as a comma separated values (CSV) file to a particular location
2. In the command line type `vu_qct` to load the tool, (see Figure 3)
3. Users are then presented with a screen where they can type in the study directory location before hitting ENTER
4. The SAS program is then run and an Excel spreadsheet detailing the inconsistencies found between the QC tracker, TFLs and datasets is produced. As well as a series of tables summarising the status of the outputs and datasets

The Bash shell script user interface allows users to tailor the tool to run in a specific study area before the script calls the SAS program, without having to update the code itself. The Bash script saves the details entered by the user as Bash variables:

```
echo -n "Input study folder (e.g. /comp100/study10/primary_delivery/): "
read path_to_study
```

and then passes these to the SAS program by using the `-set` option that is available when running SAS programs in batch mode:

```
sas path/to/file/vu_qct.sas -set study_folder $path_to_study
```

These can then be picked up within the SAS program by using `%sysget()` macro to set macro variables:

```
%let study_folder=%sysget(study_folder);
```

This way, the code can be stored in a central place and can be version controlled to reduce the risk of the code being accidentally changed. In addition, before running the code, the script includes checks to ensure that the CSV files required are present in the correct location. Often users will run this several times in the same study area until there are no errors; therefore, the script includes code that retains in memory the last area in which the program was run in, so that users do not have to provide this information for each run. This is done by creating a hidden config file in the user's home directory containing the study information:

```
echo "$path_to_study" > $HOME/.vu_qct.config
```

To reduce the amount of file space taken; the script creates a new folder to store any temporary files created during the running process. The last step in the code is to delete this folder so that the only output created is the Microsoft Excel spreadsheet which contains any errors found in the QC tracker. If no errors are found, then the spreadsheet will only contain the summary tables.

READING THE DATA INTO SAS

Before any checks between the QC tracker and TFLs can be performed, the information from the QC tracker to be checked needs to be read into SAS. The `INFILE` statement is used to read in the two QC tracker CSV files, one for the ADaM datasets and the other for TFLs, which are saved as SAS datasets. In addition, information on program names and run dates/times from the file system needs to be read into SAS.

When SAS programs are run in batch mode in Linux, two files are automatically created: an LST file containing what would go to the output window in SAS along with a LOG file containing what would go to the log window in SAS. It is possible to extract the most recent date/time of files in SAS (see next paragraph) which can be used to determine when a program was run or saved. Since it is possible for a QC program to be unchanged and re-run, in which case the QC program save date/time will be before the QC LOG file run date/time; therefore, the run date/time from the LOG file is used. For ADaM datasets, the latest run date/time is taken from the production program LOG files and for TFLs the latest run date/time is taken from the output file. The ADaM dataset run date/time and TFL run date/time for production and QC are obtained using SAS external file functions.

The `dopen` SAS function is used to open the file directory containing the TFLs and creates an identifier value for the open directory. Then the `fopen` SAS function opens the files within the open directory, using the identifier value created from the `dopen` SAS function and creates a separate identifier value for each file in the directory. Then the `dread` SAS function is used to obtain the name of each file identified by its unique identifier value within the directory and the `finfo` SAS function can be used to obtain system information such as run dates/times. This creates another SAS dataset containing the TFL program names and the latest run dates/times. These steps are repeated for the directories containing the ADaM datasets and QC LOG files for datasets and TFLs.

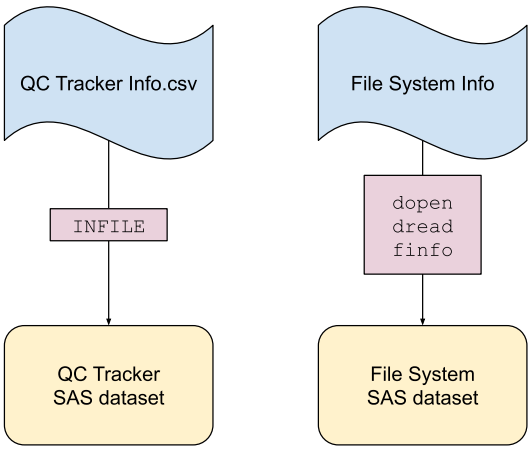

Figure 1: A graphic of the process of reading the data into SAS

CHECKS THAT ARE PERFORMED

The following three checks will focus on the accuracy of TFL information but are also be applied to the ADaM datasets. It is possible that not all the outputs or datasets have been created at the time the QC tracker check macro is run, therefore only outputs and datasets that have a date in the QC tracker are checked.

1. Check that the program name information in the QC tracker is correct

To check for typos which may have occurred in the QC tracker, the File System SAS dataset containing TFL production program names and run dates/times is merged with the QC tracker SAS dataset, by production program name. This should be a one-to-one merge since the program name is unique to each TFL. The run dates/times for the File System SAS dataset will only merge on if the program name on the file system matches the program name in the QC tracker. Only records that exist in the QC tracker are included, therefore records where the run date/time is missing indicate that the program name in the QC tracker is incorrect. The SAS code below creates a flag if the production program run date/time is missing but the production date in the QC tracker is not missing. A similar process is applied to the File system SAS dataset containing TFL QC program names.

```
if prod_datetime = . and tracker_prod_date ne . then

flag_1= "Production program name in the tracker is "||program_name||"
        but the actual name of the program is different, please update the
        tracker.";
```

2. Check that run dates match the QC tracker

Using the same SAS dataset created from the merge above, we can check that production run dates/times have been correctly captured in the QC tracker. The run dates/times obtained from the file system and the run dates from the QC tracker are converted to SAS dates before running the following SAS code. The same check is performed on the QC run dates/times. A flag is created if a mismatch is found.

```
if tracker_prod_date ^= . and prod_date ^= . then
  do;
    if tracker_prod_date ^= prod_date then

    flag_2="Production date in the tracker for program "||
           program_name||" is "||put(tracker_prod_date,date9.)
           ||", but this program was run on "|| put(prod_date,date9.)
           ||".";
  end;
```

3. Check that QC run dates/times are after production run dates/times

To perform this check, the QC tracker SAS dataset and the File System SAS datasets containing TFL QC run dates/times and TFL production run dates/times are merged together. Then the following SAS code is run to check that the QC process has been followed correctly. Note that this check is not checking the accuracy of the QC tracker but the QC process itself by using the run dates/times of both the production and QC programs from the file system.

```
if prod_datetime ^= . and qc_datetime ^= . then
  do;
    if qc_datetime < prod_datetime then

    flag_3="Program "||program_name||" was run on "||
           put(prod_datetime,datetime18.)||" which has a later date than the QC
           program "||qc_program_name||" which was run on
           "||put(qc_datetime,datetime18.)||".";
   end;
```

OTHER CHECKS

Dependencies: For CDISC studies it is necessary that ADSL is run before any other ADaM dataset. Using the File System SAS dataset containing the ADaM production program run dates/times, the following SAS code can be used.

```
if . < prod_datetime < adsl_datetime then

flag_4=upcase(program_name)||" was run on "
       ||put(prod_datetime,datetime18.)||" which is before the rundate of
       ADSL.";
```

The final check that is performed is to check that TFLs have been run after the ADaM dataset(s) which were used to create them have been run. Here, the `GREP` command is used to search the TFL program LOG files for any references to an ADaM dataset, this is stored as a TXT file along with the name of the program, so that it can be converted into a SAS dataset using the `INFILE` statement. Then the corresponding File System SAS datasets containing ADaM dataset run dates/times and TFL program run dates/times are also merged on, so that the run dates/times can be compared. In the following SAS code, the variable named “whichadam” contains the ADaM dataset which was used to create each TFL.

```
if . < prod_datetime <= adam_datetime then

flag_5=program_name||" was run on "||put(prod_datetime,datetime18.)||",
       which is before the rundate of "
       ||whichadam||"("||put(adam_datetime,datetime18.)||").";
```

After all the checks have been performed, the flags created from each check are combined into one SAS dataset for both TFLs and ADaM datasets.

Project management: Aside from performing checks, the QC tracker checker macro is also used to show how a study is progressing. It is sometimes difficult to clearly see from the QC tracker how many outputs have been created, passed QC or are yet to be started, so a series of summary tables are created which present this information in a concise way.

The final step is to output all this information into a Microsoft Excel spreadsheet using a series of ODS statements so that it can be viewed and interpreted more easily. We will now go through an example.

EXAMPLE

Figure 2 shows an example of a QC tracker in the form of a Microsoft Excel spreadsheet which includes information on TFLs and ADaM datasets. The errors highlighted with blue squares are an example of human error which can be difficult to identify. We can see that the program name for the Summary of Adverse Events, has an extra space between "ae" and ".sas" and the QC date for the Summary of Demographic Characteristics is showing the incorrect year.

The next set of errors are QC process errors and can be seen in the red squares. The first of this type of error shows that the Summary of Important Protocol Deviations table has been run after it was last QC'd. Another error that can be seen is that the ADaM dataset ADSL looks like it has been run after ADAE. The final error shows that ADVS has been run after the Summary of Vital Signs table. It is possible that the dates have been incorrectly entered by the production/QC programmer and so would be classed as human error and is more easily fixed, however if the dates entered in the QC tracker are correct then it is important that we rectify the error in the QC process.

Display Name	Display Number	Production Programmer	Program Name	Production Date	QC Comment History	QC Programmer	QC Program Name	QC Status	QC Date
Summary of Demographic Characteristics	1.0	AB	demog.sas	06-Jan-19	No comment.	BC	qc_demog.sas	Pass	09-Jan-18
Summary of Adverse Events	2.0	AB	ae .sas	14-Jan-19	Update footnote.	BC	qc_ae.sas	Fail	20-Jan-19
Summary of Important Protocol Deviations	3.0	AB	dv.sas	14-Jan-19	No comment.	BC	qc_dv.sas	Pass	11-Jan-19
Summary of Vital Signs	4.0	AB	vs.sas	13-Jan-19	No comment.	BC	qc_vs.sas	Pass	20-Jan-19
Listing of Adverse events	5.0	AB							

Dataset Name	Production Programmer	Program Name	Production Date	QC Comment History	QC Programmer	QC Program Name	QC Status	QC Date
ADSL	AB	adsl.sas	03-Jan-19	No comment.	BC	qc_adsl.sas	Pass	05-Jan-19
ADDV	AB	addv.sas	13-Jan-19	No comment.	BC	qc_addv.sas	Pass	20-Jan-19
ADAE	AB	adae.sas	02-Jan-19	No comment.	BC	qc_adae.sas	Pass	04-Jan-19
ADVS	AB	advs.sas	14-Jan-19	No comment.	BC	qc_advs.sas	Pass	16-Jan-19

Figure 2: TFL information and ADaM dataset information in the QC tracker; human errors are shown in blue squares and QC process errors are indicated with red squares and connecting lines

RUNNING THE MACRO

The QC tracker will often contain TFL and ADaM dataset information in separate tabs of an Excel spreadsheet. The first stage is to save each of these files as CSV files so that they can be read into SAS. We can then start the process of running the macro.

As shown in Figure 3, "vu_qct" is typed on the command line which prompts the start of the process.

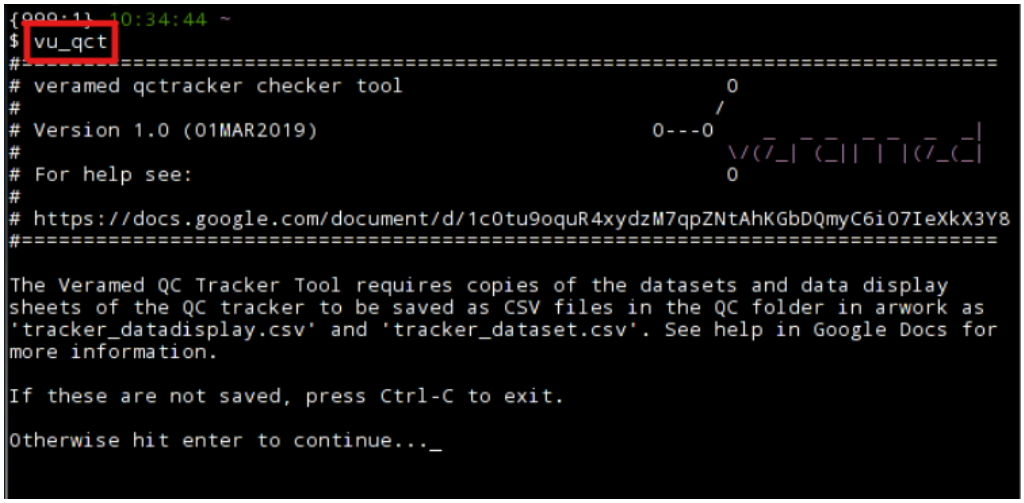

Figure 3: Screenshot of the Linux command line showing the start of the Bash shell script after typing "vu_qct"

After hitting ENTER, users are presented with questions to identify the study directory location in which the macro needs to be run in. If a user has run this code before, then this location will not have to be re-entered. The final ENTER sets the SAS program running.

VIEWING THE OUTPUT

After the QC tracker checker macro is run, a Microsoft Excel spreadsheet is created so that the user can easily see and fix any errors which were identified by the tool. The spreadsheet is made up of several tabs containing summaries of the study progress, errors found in the TFLs and errors found in the ADaM datasets. Figure 4 shows the summary tables that are created which is useful if there are many study outputs that need to be produced. It includes information about how many outputs have been created and passed QC, giving users a better understanding of how the study is progressing. Figure 5 and Figure 6 shows the errors which were found based on the example QC tracker in Figure 2. As we can see, the macro has correctly identified the typos in the dates and program names as well as the errors in the QC process. We have included as much information in these flags so that the user can clearly understand what needs to be updated.

Once most outputs and ADaM datasets have been created, this tool will be run regularly until no errors in the QC process or QC tracker have been found, saving us time from having to perform these checks manually.

Status of datasets	Frequency
Total	4
Produced	4
Not Produced	0
Stage 2 QC Pass	4
Stage 2 QC in progress	0

Display Type	Total	Produced	To Produce	Stage 2 QC Pass	Stage 2 QC in Progress	Stage 2 QC not started
Table	4	4	0	3	1	0
Listing	1	0	1	0	0	1
Total	5	4	1	3	1	1

Figure 4: Summary tables of the study progress as created by the tool

Display Title	Program Name	Production Program Differences	QC Date Differences	Prod, QC Differences	Programs run before dataset:
Summary of Demographic Characteristics	demog.sas		QC date in the tracker for program qc_demog.sas is 09JAN2018, but this program was run on 09JAN2019.		
Summary of Adverse Events	ae .sas	Production program name in the tracker is "ae .sas" but the actual name of the program is different, please update the tracker.			
Summary of Important Protocol Deviations	dv.sas			Program dv.sas was run on 14JAN19:13:38:13 which has a later date than the QC program qc_dv which was run on 11JAN19:15:10:57.	
Summary of Vital Signs	vs.sas				vs.sas was run on 13JAN19:13:38:23, which is before the rundate of ADVS(14JAN19:09:41:48).
Listing of Adverse Events					

Figure 5: Summary of the errors found in the TFLs as created by the tool

Dataset Name	Program Name	Production Date Differences	QC Date Differences	QC Program Differences	Prod and QC Differences	Dataset run before ADSL
ADSL	adsl.sas					
ADDV	addv.sas					
ADAE	adae.sas					adae.sas was run on 02JAN19:09:41:46 which is before the rundate of ADSL
ADVS	advs.sas					

Figure 6: Summary of the errors found in the ADaM datasets as created by the tool

CONCLUSION

This paper has discussed a tool designed so that all TFLs and datasets that we produce correctly follow the QC process and ensures that this process has been documented accurately. While every effort should be made to accurately enter information into the QC tracker, we are now able to use this tool to quickly perform checks which might take longer if done manually. It also allows us to view the progress of a study in a concise way for all required TFLs and datasets helping us to successfully meet our timelines.

ACKNOWLEDGMENTS

I would like to thank Nick Cowans and Diana Stuart for their help and support.

CONTACT INFORMATION

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

Anisa Jeetoo
Veramed Ltd
5th Floor Regal House
70 London Road
Twickenham
TW1 3QS
020 3696 7240
anisa.jeetoo@veramed.co.uk
www.veramed.co.uk