PhUSE 2013

Paper PP08

Two Into One

Helen Nicholson, Cmed, Horsham, UK
Kathryn Wright, Cmed, Horsham, UK

ABSTRACT

As with almost any programming group there were constant demands for deliveries of different sorts, with the requests changing all the time. With the programming group spread out, not only over three countries (and time zones) but also with some of the group working on PC SAS (v9.1) and others using SAS (v9.2) via Enterprise Guide on a Solaris server it was not always easy to move resource around to where it was needed. One part of the solution was to move everyone to use the same platform and SAS version. This poster will illustrate some of the issues encountered and the solutions reached to achieve this, to unite the group SAS environment with minimal disruption to deliveries. It will highlight some of the problems and solutions when moving programs that call other software to a platform where such interactions are not possible.

.

INTRODUCTION

The SAS programming group within Cmed is split over three locations and time zones. Historically it was split down even further with some people doing statistical programming and others producing dataset and listings for the data management group. All the 'old' statistical programmers were in one location but the others were split over the sites, which meant that it could be difficult to share tasks. Added to this the 'old' statistical programmers were using SAS version 9.2 via Enterprise Guide where as everyone else was using PC SAS (v9.1.3).

The decision was taken to amalgamate the groups so that everyone could work on anything. The final decision was for everyone to use SAS 9.3 on a Linux server using Enterprise guide 5.1. This poster will discuss the actions taken to make this happen. It will not discuss the reasons behind the choice of programming environment chosen or how the choice was made

GETTING READY

Once the decision was made about the final environment work could then begin on working out what we actually had to do. We had to work out how many ongoing projects we had, would they all be transferred to the new system or would some stay on the old. As expected there was a mixture in the size and age of projects. The larger projects tended to be the old ones but that was not always the case.

It was decided very early on that we would try and migrate as many of the projects as possible to the new system, and that we would have a dedicated team that would do the migration. The migration itself was treated as a project which meant that it could be resourced in the same way that the other projects were.

We discussed with the Quality Assurance department (QA) what would be needed to prove that the migration had been successful and had been completed. We worked out what needed to be done to prove that there had been no change to the programs, the data or the final outputs (apart from date/time stamps of the files) once the move had taken place. A migration plan was drawn up with all the relevant documentation and agreed by all concerned.

In order to make this easier to follow a checklist was written that listed the tasks that need to be done and the order in which they were to be done. This document ended up being larger than initially planned but it gave a detailed set of instructions and included the code changes that were required.

A time table was drawn up for migrating the projects, this had to take into account current deliverables, which could not be affected, and the approaching deadline for the renewal of the SAS licences. We obviously did not want to have to renew the licence for the Solaris machine, and to keep to a minimum the number of PC SAS licences that would be required. We also did not want to delay any agreed delivery timeline because of the migration.

THE MIGRATION

Once everything was ready the migration could begin. The timetable for the moving of projects was sent out so that everyone would know when their project was moving. As the migration day approached for a project the teams working on them were kept up to date so they would know if it was going to be on the planned date or not. This was important because the priorities on projects change and they were times when adjustments had to be made to the timetable to accommodate them. On the day of migration the project could not be worked on for

anything else until the migration was complete. This meant it was important to do it as quickly and efficiently as possible.

As care had to be taken that like with like were compared a copy of the everything current on the old system was made which allowed programs to be rerun if required before it was copied to the new system. The migration team took a copy of everything on the 'old system' moved things to the new set up and checked that nothing had been altered in the moving from one area to another. The required changes were then made to the code so that the programs would work in the new area, everything was then run, the new data and outputs were then compared with the files that had been copied over to make sure that the only thing that were different were the date/time stamp on (and in) the files.

This meant that we did two comparisons. One to check that the actual physical transfer to the new system had not caused any differences, the other compared the output created from running programs on the new system with the output that had been created by the old system, i.e. the new system did not create any differences.

Sometime there were differences that were not just the date/time stamp on the output, which were all investigated to make sure that it was not the transfer that had caused them. They all turned out to be data related problems, either for some reason we had to use a different cut of data than the original, so we were not comparing like with like, or the dictionaries had changed and we could not recreate the old ones for whatever reason.

The comparison was done electronically, using commercial software designed for this type of work, to make sure that it was done fully. Each time a comparison was made the outputs from the comparison were saved to document what had been done. There was a sign off sheet for each project to fulfil the QA side of it, and a report was written at the end to document what we had done.

PROGRAM ISSUES

As some of the group had already made the move from PC SAS to using a UNIX type environment a few years ago we thought we knew a lot of the changes that would be required.

We knew we would need to change the slashes in the path names, and would no longer be able to use drive letters in them, and hoped that this would be the majority of the changes for the PC code that was being moved. We also knew we had to check that there was no PC specific options in the code – so we had to make sure that all the NOXWAIT options had been removed.

The PC programs had a lot of path names for all the various directories that were used, however they had all been set up to use macro variables that were set in the start-up program. This meant that the majority of the changes would need to be made in this program with limited changes in all the other programs – the main one being to check the options as mentioned above.

It is only when going through something like this you realise quite how many macro variables have been set up all that resolve slightly differently. It turned out that some of the macros were resolving to the same final result but getting there via different route. This caused confusion to start with but once it was worked out which was which we were able to simplify some.

On the other hand because we were moving to a different environment with things now placed in different locations we can to create some new macro variables to point to these new locations. We probably ended up with the same number of macro variables in the end.

With the code copied from the Solaris machine we only really expected to have to change the logical that pointed to the correct 'home' area – so instead of using "home/samba/biostats2" we needed to use the new path "home/samba/jdrive" and where we accessed our own area from "/export/home/&usern." to "/home/CMEDGROUP/%sysget(user) ". These changes were required because of the way that the new environment was set up.

The other change we had to make to code from the Solaris was due to the fact we were changing SAS version and not due to the platform migration. We produce a lot of 'in-text' tables where the layout of the output is based on a template. We had to create one new template and then point the code to use that template rather than the old one.

When producing graphs again we have had to change a lot of the GOPTIONS slightly so that they look the same as they did before. This included changing the device and the default font. Luckily all the graph code came from the Solaris machine so the changes were very standard to make.

Neither of these changes were required on the code that came from the PC's because of the difference in the purpose of the code, The PC code produced output mainly for data management reports whereas the Solaris code was the code used to the statistical reports.

UNEXPECTED ISSUES

We thought we would be able to predict most of the issues however were aware that we would not be able to specify all the changes/issues until we started the migration, the following are some of the unexpected issues that we found.

We thought that apart from the logical change at the start of the libname statements all the code that had worked on the Solaris server should work on the Linux server, so were surprised when some code failed. We found that the Linux server is much more case sensitive than the Solaris – we had to get the whole path to match the caseexactly.

We had some unexpected results that were due to the data – these issues are explained further on.

SYSTEM ISSUES

A lot of the PC SAS code interacted with the applications on the PC to produce outputs in certain formats. With the final setup chosen these interactions were not possible so changes had to be made to the way the programs ran.

A lot of the PC SAS programs produced output for data management. The code used to open Excel from within the SAS program (using the X command) and do the formatting of the sheets using an Excel macro. This was done because the people looking at the outputs were comfortable using Excel to review data. As SAS was now running on the Linux server which did not have Excel on it this was no longer possible.

Instead the temporary solution was to get the code to write out to a CSV file and then the user, from within the windows environment, opens up a spreadsheet that runs a series of macros to read in the CSV files and produces the Excel files with the required formatting in the format people are used to. This is obviously slightly more 'long winded' and manual than before but the increased speed of the server makes the overall time shorter.

The more long term solution is to re-write the code to output directly to a formatted spreadsheet.

As the two groups had been working in separate areas, different naming conventions and directory structures were in place. All the programs were going to be run by the same people so it made sense to unify the directory structure so that anyone would know where to go without having to look it up the whole time. The resulting structure still keeps the two tasks in separate areas but under one directory tree while allowing our statistical colleagues to carry on working with only minor changes to the code.

Some of the issues we overcame by simply implementing standard rules for naming conventions within and for projects. This meant that we could write code that followed these rules for where the output was saved. As we were now working on a Linux server we had to change where some of the output was put and store it in a temporary area and then manually copy the output over to the final area when we were finished. This has the advantage of we would control when the output was ready to be seen by other members of the department much more easily.

One problem that keeps coming back to haunt is the case sensitivity of path names. Our way round this is to make sure everything is in lower case but it is surprising how often this gets forgotten

DATA ISSUES

Some of the issues found were actually caused by the data itself. The new environment seemed much more sensitive to data issues, this section will discuss some of the data issues we found that had not troubled us on either of the previous systems.

One of the biggest problems we had was with missing values. Both on the PC and the Solaris missing values were well missing but on the Linux it was a different matter when reading in data from CVS files. We got spurious characters that looked like dotted capital D with a line through them. We found we had to use the compress function to remove this character.

There were other Hex characters that caused us problems when we were reading in data that had not come in from the normal route. In the end we compressed the data to remove all Hex characters.

The other place that we had difficulty was if we were copying footnotes or things from a word document. In this cases the hyphen was not what it seemed, causing all sorts of problems until we tried just re typing the hyphen.

In all of the above the solution was simple in the end but it all takes times working out what the issue is and finding the solution he first time.

SOLUTIONS

The main solution we came up was to have dedicated team that transferred the programs to the new system. The actual program changes were basically the same for each study. More changes were required on the programs that moved from PC SAS than from the Solaris machine. The team copied the programs over, (checking nothing got corrupted in the copying over), made the changes required and re-ran the programs. The outputs were then compared against the original outputs to check that only expected changes had occurred.

Any change that was not expected was investigated to make sure that it was a valid change. We accepted changes in the date/time stamp on the outputs and any known formatting changes caused by the different operating systems.

SIX MONTHS ON

Just when you think that you have sorted everything out something still comes up to remind you that you have changed. Programs that you have not had to touch before now need to be re-run which means that you have to re-learn the changes you made at the beginning. Writing programs from scratch is fine, but every time we have to run an old program it is fingers crossed that it works and time to refer back to the notes on what to change.

In all almost seventy projects were transferred to the new system. Only a couple that interact with Excel a lot more than the others and would require a lot of manual intervention to get them to run fully have been left on PC SAS.