PhUSE 2010

Paper TS08

Development and Implementation of a Statistical Programming Environment in a Small Biotechnology Company

Albert Chau, Antisoma Research Limited, London, United Kingdom
David Shannon, Amadeus Software Limited, Oxford, United Kingdom

ABSTRACT

For small organisations, it is not always practical to deploy client-server technology for SAS® programming environment, due to the additional costs of IT infrastructure and resource/expertise. Often SAS is installed on individual PCs and data are accessed over the local area network. This poses several challenges in managing programming activities. In this talk we describe the development and the implementation of a bespoke application for creating and managing the workflow and various SAS files in a small organisation. The motivation and rationale from the business perspective, and the technical aspects of the application, will be discussed. The application runs on individual PCs without any additional investment on servers nor significant support from IT department. The application uses PROC SCAPROC to work out file dependencies in each SAS programs automatically. The underlying technology ensures possible scale-up in future. Finally user feedback on the implementation of the application will be presented.

INTRODUCTION

In this talk we describe the development and the implementation of a bespoke application for creating and managing the workflow and various SAS files in a small organisation. The solution is referred to as the Statistical Programming Environment (SPE).

The motivation and rationale from the business perspective, and the technical aspects of the application, will be discussed. The application runs on individual PC's and on a network, without any additional investment on servers or significant resource from IT department.

The application uses the SAS code-analysing procedure SCAPROC which automatically works out the input and output files for each SAS programs. The underlying .NET technology ensures possible scale-up in future if necessary.

Finally user feedback on the implementation of the application will be presented.

CHALLENGES AND BUSINESS REQUIREMENTS

For small organisations, it is not always practical to deploy client-server technology for SAS programming environment, due to the additional costs of IT infrastructure and resource/expertise. Often SAS is installed on individual PCs and data are accessed over the local area network. Many of the SAS programs are generated as a result of exploratory data analysis or ad-hoc analysis.

This poses several challenges which are discussed below:

Relationship between input datasets, program codes and outputs

For each output, there is no simple method to trace back which program and which datasets have been used to generate it.

When an input dataset has been updated or revised, it is difficult to assess which programs need to be re-run in order that the outputs are updated as a result. By manually selecting the programs to be re-run, there is a risk that some of the outputs are not updated even though they rely on the revised input dataset, and there is no easy method to identify whether all the programs that use the revised input dataset have been re-run.

Overwriting of Files

Often when a program is re-run, the outputs and SAS logs are over-written, unless the files are copied to another location. However, if the input datasets and any other programs/macros that are called within the program have not been copied over, it may not be possible to reproduce the results or outputs as these files may have changed over the course of time also.

Detection of Program Log Issues

While SAS log can be useful in detecting various programming errors, warnings and other potential programming issues (such as merging of data without BY statement etc), users often have to either scroll through the log to search for these messages, or run separate scripts to detect these program log issues. There is no quick automated way to search for errors, warnings and other notes messages and alert the users.

Running a batch of programs

Typically in a study, many SAS programs are written for different purposes, in addition to the programs used for final statistical analysis and reporting. For example: data listings for medical review, programs for interim data safety review, etc. User-specific scripts can be written to manage this in Windows environment, however this would involve users learning a different scripting language.

Documentation

There is no easy way to document the interdependencies of the various files used and generated by SAS, nor to demonstrate the outputs have been generated with problem-free programming codes.

To work out the dependencies between the input (usually SAS datasets or external data files), SAS program and the output (which can be a physical output such as PDF or RTF, or SAS datasets or other types of data files), often it requires either a programmer to document such information (in separate documentation eg Excel, or in specific format within each SAS program), or a separate script/program to scan through each SAS program. The former method is prone to human error, while the latter method only documents the dependencies between the input, SAS program and output, but it is not able to work out which SAS program(s) need to be re-run if input is changed or if outputs need to be updated.

Tools to promote standards

In order to gain efficiency, it is necessary to implement various standards that would fit in with the company processes and ways of working. These standards range from folder structure to the method of generating statistical outputs, and they need to integrate well within the proposed solution.

BUSINESS REQUIREMENT

Rather than looking for different tools or solutions to deal with each of these challenges, Antisoma chose to have one single integrated solution that would address these issues and meet the needs for the small size of the organisation, and yet have the capability for scale-up should the need arises. The solution needs to be easy to use and requires little or no additional IT maintenance.

TECHNICAL SOLUTION

The business requirements discussed above evolved into a set of technical requirements for implementation. The following list broadly summarises the functional units for development:

- **Workflow**: enable a list of SAS programs to be batch submitted and generate the SCAPROC log, whilst controlling order of execution and displaying various attributes about each program including:
    - provide a visual indicator when a log contains errors, warnings or other used defined messages
    - provide visual indicators when an input or output has been modified;

- **Batch Lists**: allow individual lists of SAS programs for batch submits to be managed;
- **Snapshots**: create secure archives of SAS programs, logs, outputs and inputs that allow future recovery;
- **Audit Trail**: document the input and output files from each SAS program, allowing the source and impact to be traced of any SAS program and between SAS programs in the project;
- **Productivity**: This refers to features that included:
  - Derivation of SAS TITLE statements
  - Derivation of SAS ODS statements
  - Directly editing SAS programs in a Display Manager Session
  - Access to SAS log error and warning messages
  - Creating of project folders to company standards

This functionality would be deliverable through Windows programming languages. As important to simply delivering was the need to ensure SPE would be a productive, not obstructive, environment for programmers and statisticians.

THE SOLUTION

SPE would be installed to a number of end users Windows desktops with the minimum of effort from IT. A desktop application written in Microsoft's .NET programming languages was chosen as the solution. This allowed robust programming within a rapid development environment for its developers, and allows IT to deploy a single Microsoft Installer Package to desktops.

The solution is constructed of components written with C# and Visual Basic. The tasks performed by the solution are fundamentally summarised with Figure 1, below.

Figure 1: Process Flow

Microsoft standards for implementing user interfaces ensured that SPE's statisticians and programmers at Antisoma were provided with an intuitive working environment that required no formal training, other than working through a self-learning guide, or a few minutes of e-learning modules.

Batch Submitting

Batch mode provides an encapsulated SAS session for SAS program. This removes the possibility of other programs temporary data sets, macro variables, formats etc. adversely affecting the current program.

Each SAS program is submitted in its own batch SAS session. An operating system *process* object is used to launch SAS in a windowless batch session. This allows SPE to track events from the SAS session, such as when it started and terminated.

Launching SAS in batch mode allows system options to be used that ensure the entire SAS log is available for SPE to read. The solution checks for errors, warnings or other messages even if the SAS program contains Proc PRINTTO statements that direct the default log elsewhere.

Impact Analysis

Core to the business requirements is the need to trace the inputs and outputs used by each SAS program. These may be SAS data sets, SAS catalogs, text files, database tables, Word documents, or any other resource that a SAS program creates or relies on to execute correctly. As SPE is deployed with SAS 9.2, Proc SCAPROC is used to gather this information.

The "SCA" in Proc SCAPROC stands for SAS Code Analyser. It requests that SAS collects numerous attributes about the inputs and outputs of both internal and external files. This information is saved to a log file, alongside the standard SAS log.

An algorithm based on graph theory was developed to calculate the interdependencies between programs and this information is stored within the solution. By querying this data the interface can flag when a dependency has

changed that requires one or more SAS programs to re-run. It also allows flags to be raised when an output has been modified outside the environment, for example, a Word document generated by SAS was opened and re-saved by another user.

The following sections describe the implementation and benefits of the technical requirements.

WORKFLOW

Building the list of SAS programs is a relatively simple affair with Windows programming languages. Algorithms were developed to recursively search folders, evaluating the files found against any programs previously used in that location.

This information is stored within a database that for each project. That database is stored as XML therefore supporting open standards and the future ability for a third-party programmer to use the information within.

The workflow is presented similarly to Windows Explorer in Detail view. It displays when a SAS log contains errors, warnings or other used defined messages. Indicators are displayed when an input or output is modified without the program being re-run.

This dependency data is used to create a report of all up and down workflow impacts. The impacts tracked are changing program inputs, outputs and the programs themselves.

The benefit to programmers and statisticians is that little effort is needed to see ensure all reports are updated when a source table changes, or determine the source of a reports contents when queries are raised.

BATCH LISTS

A single study could contain hundreds of SAS programs. Those programs may have different reporting functions such as an interim analysis, investigating ad-hoc questions or the final study report.

Consequently there is a requirement to group different SAS programs together for execution. This was delivered through a virtual folder structure called batch lists. A program from the study can be placed into one or more batch lists. Each batch list can be individually controlled with respect to the order items are submitted.

SNAPSHOTS

A snapshot creates a secure archive of all the objects in a project including: SAS programs, logs, outputs, inputs, etc. The archive needed to be recoverable without disturbing the current project contents. It also needed to be secure, so zipped archive files were created and restored via an operating system programming object that allows their creation, password protection and recovery.

The archive creation and recovery is controlled by users through a dialogue that confirms actions and the contents of zipped archive files. An internally generated password is used to ensure the archive file contents are readable, but never editable.

AUDIT TRAIL

An audit trail of program dependencies, the status of each program and dates information can be exported to a Microsoft Excel workbook.

This reports various attributes of each file in the SAS project, such as for dates created, modified, and for SAS programs when submitted and by who. A second list detailing the inputs and outputs of each process was also presented.

The solutions database contains the information required for the audit trail. This is collected and refreshed after each batch job completes by harvesting information from the SCAPROC and standard SAS logs. Producing the reports required a series of queries with Microsoft's Language Integrated Query (LINQ) extensions for C# to derive the required report structure. Exporting the information to Excel required a further Windows programming object to create and maintain Excel Workbooks. This functionality is triggered by the user from a pull-down menu option.

PRODUCTIVITY FEATURES

Additionally, the following features were integrated into SPE with the intention of removing repetitiveness and the need for human intervention in the programming and reporting process.

Defining SAS TITLE Statements

Project teams use an Excel workbook for, amongst other reasons, the planning of table, listing and figure titles. The solution uses this information to populate macro variables in the appropriate SAS programs. In turn these are resolved into title statements through SAS macros.

This feature removed the need to define titles for each SAS program, instead a call to a standard SAS macro is used which promotes portability between programs.

Deriving ODS Statements

A default report document type was desired; however the flexibility to alter this for any SAS program was also needed. Potentially many output files could be created at once, such as Excel, HTML, PDF, RTF or the traditional Listing file for each program.

SPE provides an easy method for users to select one or more output types for a SAS program. These values are used to build one or more ODS statements via a standard macro call. An added benefit is the ability to track what titles actually appear in each open ODS destination. Mostly titles will be defined from an Excel spreadsheet as discussed above; however occasionally this may need to be overridden.

The following query is used to collect and report the actual titles found in each report:

```
proc sql noprint;
    create table _odstitles_ as
    select distinct xpath, dest,number, text
    from dictionary.extfiles e,
         (select case upcase(destination)
            when ("LISTING") then "LST"
            when ("TAGSETS.EXCELXP") then "XLS"
            when ("HTM") then "HTML"
            else destination
          end as dest
          from dictionary.destinations
         ), dictionary.titles
    where (e.directory='no' and e.exists='yes' and
           strip(scan(upcase(xpath),-1,'.'))=dest and
           type='T'
           and ^missing(text)
           );
quit;
```

By querying three dictionary tables within the SQL procedure, the open ODS destinations, Titles and External File names are combined and available for reporting.

Direct Edit of a SAS Programs in a Display Manager Session

To encourage usability within SPE, users can edit a SAS program in a display manager session (DMS) with one button press. The autoexec program is submitted ensuring that libraries are assigned. The SAS environment settings are therefore identical to when the program is executed in batch mode.

Direct Access to SAS Log Error and Warnings

The solution needed to help programmers and statisticians see what error or warning messages prevented programs from completing successfully. Often programs are long and may contain many hundreds or thousands of messages.

A log viewer was built that implemented a fast routine for opening long log files, simultaneously counting the numbers of notes, warnings and errors. Toolbar buttons allow direct navigation to the next error or warning message.

Creating of Project Folders to Company Standards

Antisoma and Amadeus defined a standard folder hierarchy for study reporting. This contains various levels for internal and external reporting, in addition to QC. The solution allows the full folder structure to be built and when needed, the structure to be overridden.

This feature promotes consistency and removes the need for intervention by deriving study autoexec programs.

CONCLUSION

We have seen that the development and implementation of a statistical programming environment has been achieved to deliver consistency, productivity and auditing for a small biotechnology company. Some of the benefits brought on by the Statistical Programming Environment include:

(1) Easy method to detect errors, warnings and custom text
After a program is run in SPE, the numbers of errors, warnings and custom text that exist in the associated SAS log would appear. In addition, the status of the program would also alert the user that the program has not been run successfully, so that the user knows whether further action on the program is required.

(2) Easy to see what has been changed
If a dataset has been modified, then the status for all the programs that read in this dataset would change to "Input Has Changed", so that users can see straight away which programs will need to be re-run.

(3) Ordering of Program Submission
The order in which the programs are submitted can be important. The SPE allows the users to have an easy method to specify the order in which the programs are run.

(4) Dependency Analysis
The dependency analysis allows data to be traced through all the SAS programs that are dependent upon it. Similarly, all the items that any output depends on can be easily traced. All the information is documented in both the SPE solution for visual use by users and an Excel spreadsheet for use outside of SPE.

(5) Snapshot Function
The snapshot function allows user to archive the files within a project folder (including all the sub-folders). The users have also identified a few areas of improvements and opportunities for further development:

Several areas of improvements have been suggested and these include:

(1) Multi-User Access
While multiple users can access the same project area and run SAS programs, only the first user has write access to the SPE project. All other users are warned, but functionalities such as detecting errors, warnings, custom text in the SAS log still work, but are not recorded in the SPE project.

This is due to the restriction within Windows environment, where only one person can have write access to a file at any time. In order to allow simultaneous write-access for multiple users, it would be necessary to incorporate a non-memory resident database for the SPE project file. While this can cause problem when there are multiple users working on the same project at development stage, it is considered not a major issue when it comes to production, as usually one person would be responsible in submitting a whole set of programs.

(2) Performance
It takes slightly longer to run a SAS program via SPE, compared to batch submit in Windows and interactive SAS. However the additional time is minimal – this is possibly due to the additional step of accessing the Excel workbook for the TITLE of the outputs.

In conclusion, the SPE has been proven to be a valuable tool to the company. It is intuitive to use and it does not take a lot of time for users to learn how to use the application.

CONTACT INFORMATION

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

Albert Chau
Antisoma Research Limited
Chiswick Park Building 5, 566 Chiswick High Road
London
W4 5YF
United Kingdom
Work Phone: +44 (0)20 3249 2100
Email: albert.chau@antisoma.com
Web: www.antisoma.com

Or

David Shannon
Amadeus Software Limited
Mulberry House, 9 Church Green
Witney
Oxfordshire
OX28 4AZ
United Kingdom
Work Phone: +44 (0)1993 848010
Email: david.shannon@amadeus.co.uk
Web: www.amadeus.co.uk