Paper 40-26

Managing the development of SAS® applications with SAS

Frank Poppe, PW Consulting, the Netherlands

ABSTRACT

The change to SAS version 8e has important consequences for the application development process. An application is no longer limited to SAS but can involve things as a broker configuration, HTML-code, CSS-files, JavaScript-code, applets, etc.
This paper investigates the consequences for an applications development management tool.

SAS always has been a system that not only provided ready to use solutions, but also the means to develop applications (from small macros to large packages).
The development of large applications within SAS calls for tools to manage the development process. These are not standard available, and therefore the 'SAS Applications Environment' (SAE) was developed. Key functions of SAE are the following:

- different roles and rights (co-ordinator, developer, etc.).
- automatic locking when an element is downloaded for editing;
- testing stand-alone or in combination with other elements;
- programming tools like catalog wide searching, neat layout of code, etc.

In version 6 an application was confined to SAS and the user interface was handled through macro parameters or through SAS/AF® or SAS/FSP® screens, etc. With the shift to thin clients now dynamic Internet output has to be produced. These developments have a great impact on applications that support the management of this type of SAS applications.

INTRODUCTION

Almost from the start SAS has been a system that not only provided ready to use solutions, but also could be used to develop applications. These applications range from relatively small macros for specific tasks, to large packages that deliver new functionality's to the users.

The development of large applications within the SAS system calls for tools to manage the development process. The following aspects typically are the case when the need for such a tool arises.

- Different people work on the application (all at the same time or spread over time);
- The application consists of several parts (which should be tested separately as well as integral);
- Different versions and phases (in production, in testing, under development) have to be kept and synchronised;
- Neat layout of code should be ensured (for future maintenance).

Such a tool is not standard available in the SAS system.

For these situations PW Consulting developed already years ago SAE. SAE operates in SAS version 6 and the arrival of SAS version 8e meant that changes became necessary. SAS applications are no longer limited to SAS itself, but relate to and work with the outside (Internet) world. In the followings paragraphs several aspects of this development will be highlighted.

So a successor for SAE had to be developed, to profit from the possibilities of SAS version 8e, and cater for the broadened scope of applications.
To emphasise the broadened perspective the application will get a new name: the current working name is SÆntral: indicating both its ancestry and its central role in the application development process.

This presentation is not about SÆntral. It is about the consequences the new possibilities of SAS version 8e have for applications that operate in an open context. In a closed context an application just does not use a new possibility if it is not catered for in the code: if long variable names are a problem in the code you just don't use long variable names. Open applications have to be prepared that the user will want to use al those new options and possibilities.

APPLICATIONS DEVELOPMENT: COMMON PROBLEMS

The development and maintenance of a large application can become a process that is difficult to manage:

- a growing number of people is working on the application,
- different version have to be maintained,
- in production the application is spread over different machines,
- etcetera.

In practice several problems can be observed in these situations. If you have worked on such applications (or worse: have been responsible for them…) you will recognise one or more of these problems.

INTERFERING WITH EACH OTHERS WORK

Take for instance two developers who accidentally have been working on the same module. When they finish they have either produced incompatible changes, or one of them has overwritten the changes of the other. Possible causes are that they did not know that each of them needed to make changes in that particular module for the problems they were working on. Or one on them forgot some time age to refresh the common storage location with a changed module.

GOING BACK TO A PREVIOUS VERSION

Unless you are very lucky, or a very good programmer, you will at one time have applied a change to a piece of code to resolve a bug later proved to have unwanted side effects elsewhere. The old version of that piece of code has to be reproduced immediately, but that is not always so easy.

WORKING ON DIFFERENT VERSIONS

A team of developers is working on the next version of an application. At the same mean time a bug report makes it necessary to change, test and put again into a production a module of the existing version of the application, without interfering with the development process.

WORKING WITH DIFFERENT PLATFORMS

Apart from the shift to web-enabled applications (which will be treated in following paragraphs) there already has been a trend to share the workload between different machines. In these situations the main application may run on your PC, while parts have to run somewhere on a server (either synchronised within the application, or asynchronously and perhaps in batch mode). This means that as a developer you have to move different parts to different destinations when you go into production.

FROM VERSION 6 TO VERSION 8E: APPLICATIONS HAVE TO CHANGE TOO…

GENERAL SAS ENHANCEMENTS

In SAS version 8e a number of seemingly small enhancements has been introduced: long names and labels for SAS files, catalogs and variables. The fact that the SAS users had to wait until this version of SAS for this enhancement (common under Windows since 1995), perhaps already indicates that programmatically this is not easy to implement. In an existing SAS application, that has to take these changes into account in all possible places, a lot of changes have to be made.

THIN CLIENT & WEB-ENABLEMENT

For a long time a SAS application was something that worked only with or within SAS. Particularly the interface with the user was handled e.g. through macro parameters or through SAS/AF or SAS/FSP screens.
Since the development of the web users now are saying that they want to access their application through the Internet, and you as a developer have to make that happen. This may be done by delivering the complete fat-client application with all its bells and whistles through browser windows (using products like Tarantella®). This would only require a simple installation process on the server side, using the standard Internet browsers on the client side.

But especially when you have to develop a new application, or have to make a complete overhaul of an old one, you will be confronted with the task to build a web-enabled SAS application.
The interface no longer will be a SAS/AF screen or something like that, but will be a combinations of static and dynamic HTML files, with Cascading Style Sheets, JavaScript files, JavaApplets, etcetera.

So, one of the consequences of developing a web enabled SAS application is that non-SAS files become an integral part of the application. Not only consistency between SAS files and SAS catalog entries has to be maintained, also a wide range of non SAS files have to be kept under control: the application has to be capable to 'reach outside SAS'.

THE ELEMENTS OF AN APPLICATION

The integration into the application of activities taking place on other machines (SAS-server, web-server) also means that the application has to handle elements that eventually will reside on different machines. And not only will the number of elements grow rapidly, also the number of different types will grow.

Some elements will belong together (like the SCL entry can be linked to a FRAME or CLASS entry, and a number of JS-files and CSS-files have to reside in the same place).

To be able to handle all these files comfortably, it will have to be possible to group the elements logically. Then the group as a whole can be managed. This 'management' concerns the different actions (upload, download, edit, delete, etc.). Which functions are actually implemented will depend on the type of element, and may differ between 'single elements' and groups defined in the sense above.

The implementation of these actions will be in such a way that new element types can be added easily. The application has to 'know' what action has to be taken when a developer wants to edit an element: open the build window of SAS/AF, or open an externally defined HTML-editor, etc.

MORE DISCIPLINES: MORE (DIFFERENT) PEOPLE

One of the main purposes of the application is to make it possible that different people work on the application. Traditionally a limited number of roles has been defined:

- the developer, with limited right within a specific application he is assigned to;
- the co-ordinator for an application, who has all rights within the application;
- the supervisor, with complete control across all applications.

But when the scope of the application grows, so will the number of disciplines working on the application. The SCL expert will not necessarily be the same person as the SAS/GRAPH® expert, and JavaScript may involve yet other people.
This means that it will be necessary to have a flexible scheme of defining and assigning roles and rights to persons.

THE USER PERSPECTIVE

The only good way to show how the user will experiences an application as described above is to demonstrate the prototype. That cannot be done here on paper. Instead I will try to describe what you should expect from such an application.

A TYPICAL SITUATION

Let us first describe a typical situation.

Several applications have been defined and for each project a person is designated as the supervisor.

The supervisor defines the elements of the application (SAS catalog entries, SAS data sets, other files). They do not have to exist yet in reality, although he can also create empty templates or import elements from outside. In other words, the system uses meta-data. This means it will not operate on all files that are in a particular directory, but will only take into account the files that have been defined as elements of the project.

The supervisor for an application also assigns the roles and the rights of the developers working on the project. Some developers may e.g. get the right to create new elements within the application. If a developer does not have that right he can of course physically create elements that have not been defined by the supervisor, but they will exist only in his own environment and he will not be able to upload them (until the supervisor perhaps defines it).

THE DEVELOPMENT PROCESS

The application takes care of the status control on the elements of the application. Initially they are all locked. The supervisor (or maybe a developer who has got that right) can release all or particular elements for editing, for all or for particular developers. A developer then can ‘download’ an element for editing and it will then be locked for others; but they can still view and use the original version.

The first action of the developer may be the actual creation of the element. In later stages it will involve minor or extensive editing. The developer will also have facilities to test the element or elements he has downloaded, either as stand alone elements (if that is feasible) or using other elements of the elements from the common storage area.

If a developer ‘uploads’ a changed version the changed one becomes available for the others, and the original version is stored for backup.

As a developer you can create new elements that are not defined in the meta-data. But only when the supervisor (or somebody else who has been assigned that right) defines it in the meta-data you can make it available to other people.

TEST: ACCEPT OR REJECT

If developers report that their tasks have been executed the supervisor will want to test the new version of the application. Or he wants to assign that task to other people, whose role only gives them the right to read and/or execute elements (and not to download them). Bases on the results of this first acceptance test, the supervisor has to decide what to do next. If the test result is not satisfactorily elements will probably remain open for further editing. Some elements can, if necessary, be replaced with an older version.

When the test results are positive, the work may not be finished. But if a certain milestone has been reached the current state of the application can be ‘frozen’ as a ‘version’. Later one can refer to that version in order to reload all the elements in the current state.

DEVELOPER TOOLS

The features described so far concentrate on the management of the development process. Equally important is the support of the developer. Many features that have proven themselves important under SAS version 8e, will remain important under v8.

E.g., re-alignment of code after editing and cut-&-paste actions will make the current IF-THEN-ELSE and DO-END structure clearly visible again.

Also text-searching across SCL and SOURCE entries will e.g. show where a certain method or subroutine was used.

On the other hand the possibility to force a colour scheme on an SCL entry will no longer be useful once the Enhanced Editor can be used there.

Some changes described in the previous paragraphs lead to changes in the interface. The growing number of elements to be managed, and the added possibility to group them hierarchically, logically lead to an interface where those groups can be folded in and out. Luckily SAS version 8e has better support for such an interface.

This tree (from applications through hierarchical groups down to the smallest elements) then forms the central point of the user interface. Here the developer sees the meta-data for the application, and the status of the different (groups of) elements (free, locked, downloaded, etc.), and here he can execute the actions on the elements and on the groups.

CONCLUSION

The need for an application that supports the management of the development process of SAS applications will grow now SAS applications become more complex, involving non SAS interfaces and utilities, and different platforms.

More and more flexibility will be demanded from such an application, regarding the organisation of the different elements and the roles and rights of the persons working on the project.

The system also will have to supply a clear and simple interface for the developers, also offering a set of programming tools.

The step from SAS version 6 to SAS version 8e makes it very worthwhile for supporting applications like SAE to change too (to SÆntral) to profit from the new possibilities and to make them available to the users and the applications they develop.

CONTACT INFORMATION

Any comments, suggestions or questions regarding this paper are welcome.

Frank Poppe
PW Consulting
visiting address:
Stationssingel 25, Culemborg

postal address:
PO Box 373
4100 AJ Culemborg

telephone:	+31 345 545060
fax	+31 345 518090
e-mail:	Frank.Poppe@PWcons.com
web-site:	www.PWcons.com