

What you get and what you lose when you switch to SAS Enterprise Guide®

Andreas Mangold, HMS Analytical Software GmbH, Heidelberg, Germany

ABSTRACT

SAS Enterprise Guide® has evolved over several Versions (now 4.2) and is now a widely accepted Frontend for SAS software users and it is usually included in a SAS Software license. But experienced SAS programmers often don't like it, and they have their reasons. This presentation contrasts these reasons to the extended features of SAS Enterprise Guide in order to show helpful practices:

- What are the main complaints of experienced SAS programmers about the tool and what workarounds are available?
- How can the experienced SAS programmer benefit from integrating SAS Enterprise Guide into his daily work?
- How does SAS Enterprise Guide help inexperienced users to learn SAS?
- How can an organization benefit from extended features of SAS Enterprise Guide like publishing SAS-code as a stored process or using custom tasks?

INTRODUCTION

I have been coding SAS programs for about 25 years and I know how fast programming with the traditional SAS windowing environment (which I will call "Display Manager" or "DMS" here) can be. Although SAS Enterprise Guide ("SEG" in this paper) slows down the core programming work, I began using SEG since version 3 because it helps organizing the data and the programs. This paper covers the newest version 4.2 of SEG¹ which works with SAS 9.2.

SEG and Display Manager have rather different software architectures. Display manager (Figure 1) is the GUI part of an integrated rich client programming environment which comprises data access, code execution and result delivery, which is poorly integrated into Microsoft windows but can be used on various operating system platforms.

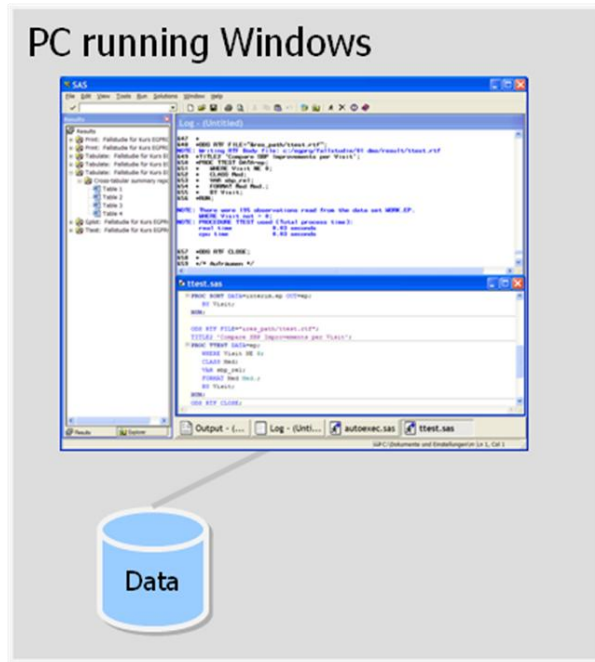


Figure 1: SAS Display manager software architecture (Windows)

¹ See <http://support.sas.com/documentation/onlinedoc/guide/index.html>

SEG (Figure 2) is a front end application with a good integration into windows but with very limited data access capabilities and without any code processing engine. SEG always needs a SAS server (at least a local one) to execute code and access data. SEG can be used as a development environment for SAS programs or as a point and click interface where SAS programs are generated in the background by so called “tasks”. Both approaches can be mixed.

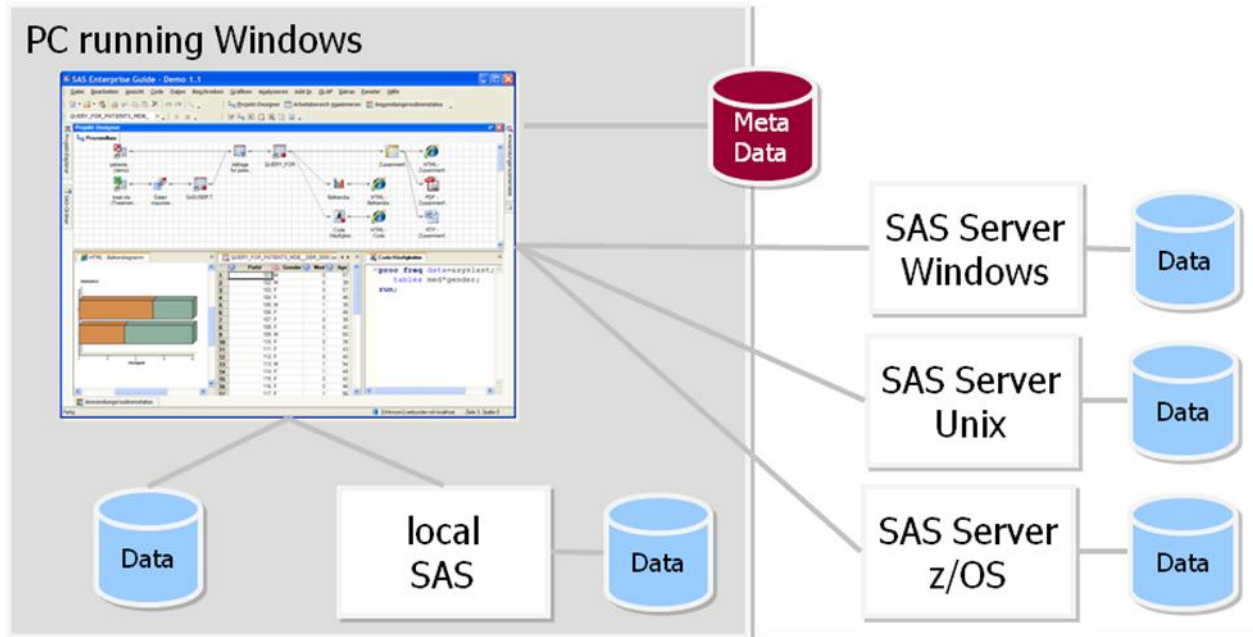


Figure 2: SAS Enterprise Guide software architecture

WHAT YOU LOSE

YOU LOSE VERY QUICK PROGRAM-SUBMIT-DEBUG-CYCLES

Due to the software architecture of SAS Enterprise Guide, some time is needed to transfer SAS code to and results and logs from the SAS server to the front end, even if SEG is installed locally. This can slow down the program development speed.

Furthermore, full keyboard control is missing in SEG. Your hand must jump from keyboard to mouse and vice versa rather often. Finally, startup of SEG is slower than DMS because of much larger executables.

YOU LOSE SOME FEATURES

The following features from SAS Display Manager are not available in SAS Enterprise Guide. Mostly, this is due to the client server approach, where frontend and backend are separate applications.

Feature	Application	Work-around
Program Editor	Editor with line numbers and line commands like under SAS for Unix	Use enhanced editor
SAS/FSP	Editing of SAS data sets with extensive data validation features	None in SEG (grid editing is not a real work around) – use different tools and data import to SEG
SAS/AF	Object oriented application development	Use prompts (see below) or custom tasks ² – Microsoft Visual Studio programming is necessary, see also page 6
Macro Window	%WINDOW-statement: macro variable prompts during program execution	Use Prompt Manager (“View – Prompt Manager”) – defines user prompts which are shown before a program runs, user supplied values are stored in macro variables
Catalog Browser	Browse through catalog entries	Download custom task Catalog Explorer ³
Graphics Viewer and Editor	View and edit graphs in SAS	Use standard graphics files and your favorite graph editor
DDE	Dynamic data exchange from and to Microsoft Excel	Where import/export tasks are not sufficient, use %XLXP2SAS for import ⁴ or ODS taglib ExcelXP for

² See <http://support.sas.com/documentation/onlinedoc/guide/customtasks/index.htm>

³ See <http://support.sas.com/documentation/onlinedoc/guide/customtasks/EG4AddInBinaries.zip>

⁴ See http://support.sas.com/rnd/papers/sugi31/excel_115.zip

PhUSE 2009

Feature	Application	Work-around
		export ⁵
X-Command	Execute OS-level commands like DIR etc.	Use SAS file-IO-functions (FINFO returns also file size and change time in SAS 9.2 under Windows)
DM-Commands	Control SAS Display Manager GUI	Use automation where possible ⁶
features specific to the machine running SAS	when SEG uses a SAS server on a different machine than DMS is running on, change of SAS programs might be necessary due to different operating system or differences in available resources	none, change programs

YOU LOSE SOME TIME FOR TRAINING

SEG users need more training (I would recommend one day) than DMS users because of the more complex user interface of SEG.

WHAT YOU GET AS AN EXPERIENCED SAS PROGRAMMER

Besides the disadvantages of SEG mentioned above, there are many advantages of SEG in the fields of program structure, traceability and documentation as well as for quick insight into data and support for difficult programming tasks.

REUSE YOUR EXISTING SAS PROGRAMS

With the exception of the topics mentioned in the “What you lose” section, every program that runs in DMS also runs in SEG. This is almost always the case if you run SEG with a local SAS session. If you do not only switch to SEG, but also from local SAS to server based SAS, different environments for data access and program execution can be an issue.

The editor window has the same functionality in SEG and DMS. It is basically the same software component and adjustments for user options are reused when both tools run on the same windows machine (for instance keyboard macros, syntax highlighting options, tab settings and so on).

ORGANIZE YOUR PROGRAMS AND RESULTS

In SEG, you can organize multiple programs in a project. In a process flow diagram, you can see which results (data, reports etc.) a program produces and you can document how the programs depend on each other. Also, the SAS log of the last run stays always associated with every SAS program in the project and can be accessed at any time.

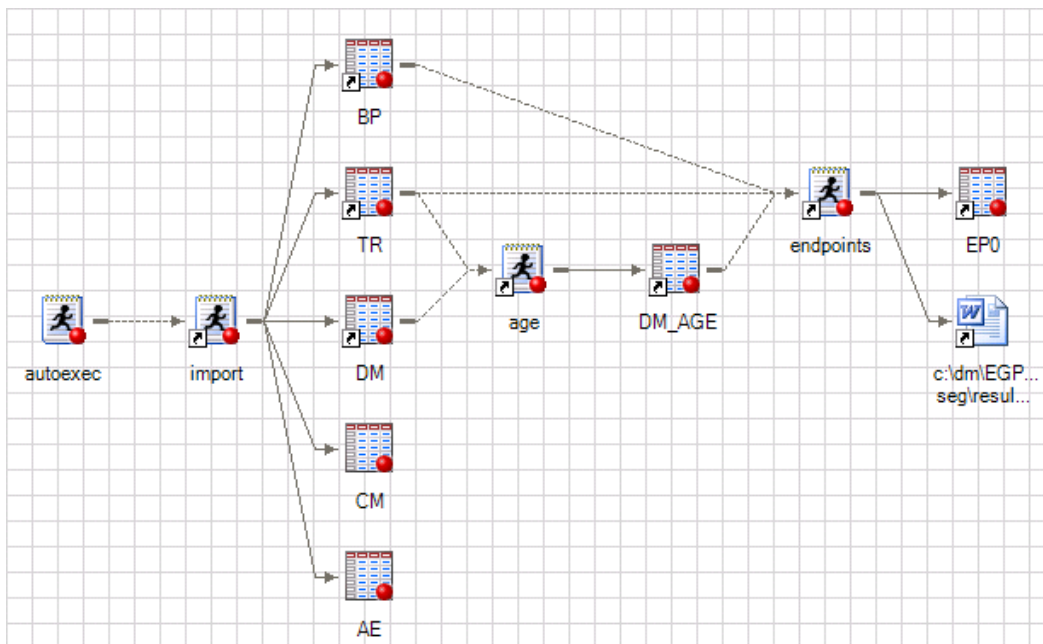
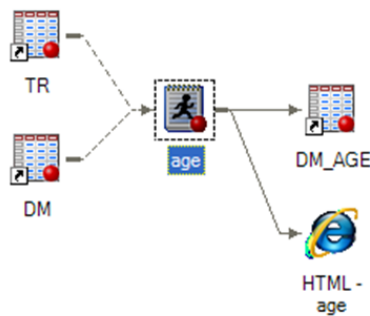


Figure 3: show dependencies of programs

In order to keep the process flow tidy, you should delete temporary SAS datasets at the end of each program as long as they will not be needed by other programs.

⁵ See <http://support.sas.com/rnd/base/ods/odsmarkup/index.html#download>

⁶ See SEG online documentation and <http://support.sas.com/documentation/onlinedoc/guide/SASEGScripting41.zip>



```

PROC SORT DATA=interim.tr OUT=tr;
  BY PatId;
RUN;

PROC SORT DATA=interim.dm OUT=dm;
  BY PatId;
RUN;

DATA interim.dm_age (DROP=DoseDate
  LABEL="Demography, age calculated");
  MERGE
    dm (IN=InDM)
    tr (IN=InTR)
  ;
  BY PatId;
  ATTRIB age LENGTH=8 LABEL='Age on DoseDate';
  age = year (DoseDate) - YearOfBirth;
RUN;

PROC PRINT DATA=interim.dm_age;
RUN;

```

Figure 4: visualize traceability

QUICKLY PERFORM COMPLEX JOINS

If your assignment is to join data from different data sets with many attributes and complex foreign key relations, this is quickly done using the Query Builder. If you are a very experienced data step programmer and you know your entire variable names by heart, coding will be quicker than clicking in the Query Builder, but not all situations are like that.

QUICKLY INSPECT YOUR DATA

When you open a data set, a whole set of tasks is available to inspect and analyze your data. Data can be filtered and sorted, all sorts of graphs and reports can be quickly generated without coding. Inspection results can be left in the project or quickly be removed.

If certain analysis steps are used often, variable assignments and options adjustments can be stored in task templates and applied even quicker.

Mixing tasks and SAS programs in a single process flow is straightforward.

	PatId	Visit	Measure	sbp_abs	sbp_rel	dbp_abs	dbp_rel
1	101	0		0	0	0	0
2	101	1		-1	-0.0064103	-6	-0.0508475
3	101	2		-8	-0.0512821	-19	-0.1610169
4	101	3		-9	-0.0576923	-24	-0.2033898
5	101	4		-8	-0.0512821	-23	-0.1949153
6	101	5		-7	-0.0448718	-22	-0.1864407
7	103	0		0	0	0	0
8	103	1		-7	-0.0426829	-13	-0.107438
9	103	2		-15	-0.0914634	-23	-0.1900826
10	103	3		-18	-0.1097561	-25	-0.2066116
11	103	4		-17	-0.1036585	-24	-0.1983471
12	103	5		-14	-0.0853659	-22	-0.1818182
13	104	0		0	0	0	0
14	104	1		-8	-0.0496894	-7	-0.0578512
15	104	2		-12	-0.0745342	-17	-0.1404959
16	104	3		-14	-0.0869565	-25	-0.2066116
17	104	4		-18	-0.1118012	-20	-0.1652893
18	104	5		-16	-0.0993789	-21	-0.1735537
19	105	0		0	0	0	0
20	105	1		-7	-0.0448718	-8	-0.0666667
21	105	2		-12	-0.0769231	-24	-0.2
22	105	3		-16	-0.1025641	-32	-0.2666667
23	105	4		-14	-0.0897436	-24	-0.2
24	105	5		-14	-0.0897436	-27	-0.225

Figure 5: quickly inspect your data

USE TASKS AS CODE GENERATORS FOR RARELY USED SAS PROCEDURES

Are there any SAS procedures you don't use often and thus do not remember the syntax? For me PROC TRANSPOSE is such a candidate. Use a task (for instance the Transpose task), run it and choose "Add as Code Template" from the context menu. The resulting SAS program is a good starting point for further work.

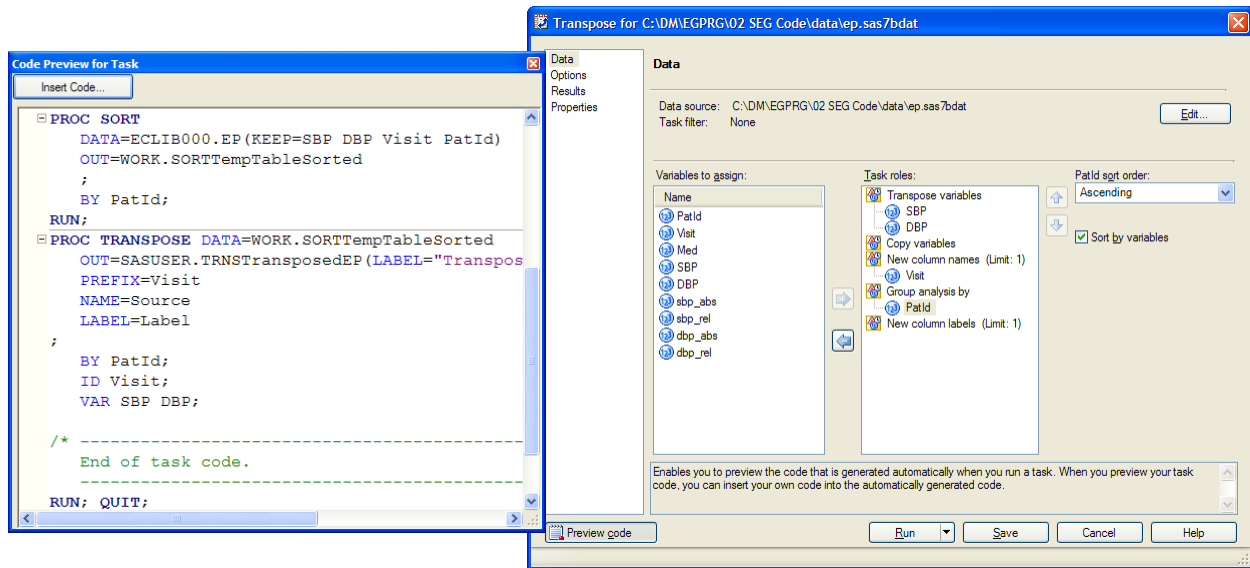


Figure 6: Use tasks as code generators

EXPORT COMPLETE PROJECT AS A SAS PROGRAM FOR PRODUCTION PURPOSES

If a process flow should be run in a different environment than SEG, a single SAS program can be generated from the process flow (or from all flows of a project). On the process flow menu, use “Export – Export all code”. Code will be exported in the correct order if dependencies are set correctly.

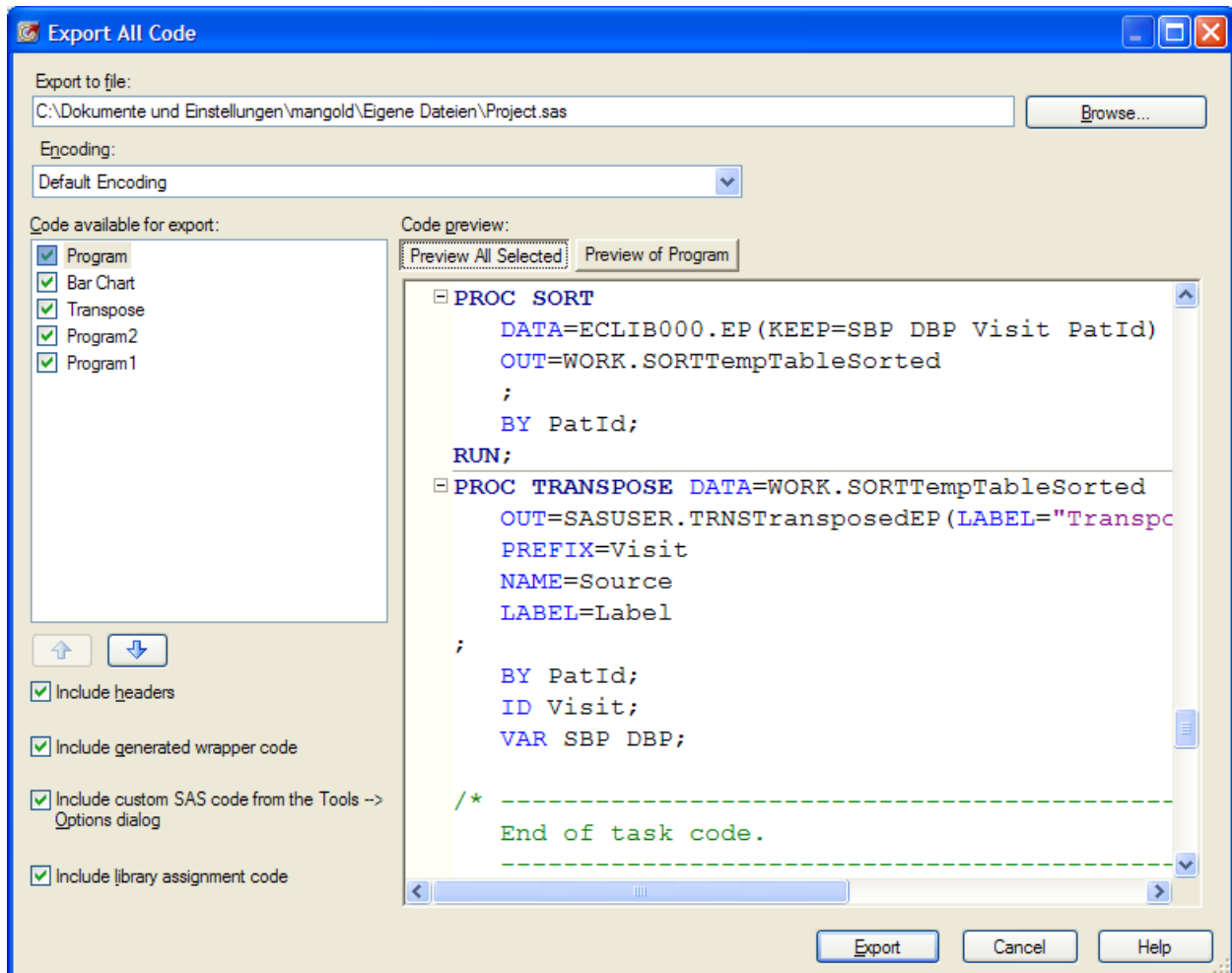


Figure 7: export all code from a process flow

WHAT YOU GET AS A SAS LEARNER

SEG simplifies learning SAS in a friendly guided environment where point and click interfaces and coding coexist.

EASY DATA IMPORT AND EXPORT

In most standard situations, data import from excel, access and text formats is very easy and guided by wizards. This has been significantly enhanced in SEG version 4.2.

The same applies to export to various formats.

OFTEN NO NEED TO WRITE CODE

Most basic functions in SEG can be used without coding. There are guided tasks for data import, data management, analysis and evaluation. Of course, this is not enough for the assignments of a clinical programmer, but it might be sufficient for quality checks and validation.

WATCH THE WORK OF THE CODE GENERATOR

Every task in SEG has a button “Preview code”. It shows the SAS code which is being generated by the task. Whenever variable roles or options are adjusted, the code changes immediately, reflecting the adjustments. This helps to gain insight into the semantics of the SAS statements and options.

This is similar to the point “Use Tasks as code generators” above.

NO NEED TO DEAL WITH ODS OUTPUT

ODS output is controlled by some basic option adjustments of SEG which can be easily changed to get different output formats (HTML, RTF, PDF, text), different styles and different graph formats. Output is handled in the background and stored permanently in the project. Traceability between data, task or SAS program and output is clearly documented. Even for SAS programs, no ODS statements have to be coded, but can be coded if necessary.

CUSTOMIZABLE USER INTERFACE

In some situation, the richness of functionality of SEG can be confusing. In many cases, some of the provided tasks cannot be used because necessary SAS module licenses are lacking. Since version 4.2 of SEG, most of the functionality can be masked for novice users so they see only the functions interesting for them.

HOW THE ORGANIZATION CAN BENEFIT

Standardization is facilitated by two advanced features of SEG.

EXPOSE SAS-PROGRAMS TO NON-SAS-USERS

If you have set up a SAS BI server environment, it is very easy to deploy any SAS program or process flow as a stored process. This can be very useful for standard database reporting where not everybody involved is a SAS user.

A stored process is a piece of SAS code on the server which can be called from various front ends, particularly from a simple web frontend. Stored processes can have parameters. When the process is called via a web interface, the user has to fill in the parameters and the values supplied are passed to the SAS program as macro variables. The generated report opens in the browser.

ENHANCE SEG BY CUSTOM TASKS

If an organization needs more complex data analysis in the context of standard business processes, SEG can be enhanced by custom tasks. A custom task integrates into SEG just like any built-in task. It has a dialog for variable roles and options adjustments and a code generation component which generates SAS code according to the user adjustments and sends it to the server.

Custom tasks can be developed with Microsoft Visual Studio and C# or Visual Basic.⁷

CONCLUSION

If you as an experienced SAS programmer can chose between DMS and SEG by yourself, your decision could be: When I work on complex macro systems, I still use the Display Manager in order to exploit its quick program-submit-debug-cycles. But for data and analysis oriented work, I use SEG because it facilitates structured work and documented results. This is even more the case in version 4.2, which has been significantly enhanced.

If you are new to SAS applications and if you can chose, I would recommend that you start working with SEG.

If your organization has to decide whether to switch to SEG from DMS, there are the issues in section “What you lose” which you should consider very thoroughly. The outcome will depend on your current SAS application portfolio.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:
Andreas Mangold
HMS Analytical Software GmbH
Rohrbacher Strasse 26
69115 Heidelberg, Germany

⁷ See <http://support.sas.com/documentation/onlinedoc/guide/customtasks/index.htm>

PhUSE 2009

Phone: +49 (6221) 6051-0

Fax: +49 (6221) 6051-720

Email: andreas.mangold@analytical-software.de

www.analytical-software.de

Brand and product names are trademarks of their respective companies.