Developing your SAS® applications the SAM way

Steve Binning, Niklas Data, Sweden
Annette Runsteen, Niklas Data, Sweden

Introduction

The purpose of this paper is to present a development environment that makes life easier for the SAS application developer. It is called SAM - SAS Application Manager, and it has been developed at Niklas Data, Sweden. In SAM you can build any kind of SAS application, online or batch. SAM provides you with what you need for professional system development, for example program control and a development, test and production hierarchy.

Niklas Data Group

Niklas Data Group is a group of SAS consulting companies owned by the founders and management. Since the beginning in 1987, our vision is to be the premier provider of solutions based on SAS. In 1994 we were the first in Scandinavia to be certified as a SAS Quality Partner. Currently, we have offices in Sweden, Norway, Finland and the Netherlands. Our head office is situated in Netherlands and during the next five years we have plans to open offices in Denmark, Belguim and Germany. We are unique in many aspects but the main one is that we only work with SAS.

Niklas Data Group employs more than 50 consultants specialized in building SAS applications for organizations and corporations. Our plans are to grow to more than 100 consultants during the year, due to the large demand of expertise in the areas we specialize in. Some examples of solutions we have helped our clients with are Data Warehouse, Executive Information Systems, Quality Statistics applications, Analysis applications, Forecasting applications and numerous different statistical applications.

Why do we need a development environment?

First of all, we need program control. If we are more than one developer working on the same application, we have to prevent the programmers from writing in the same source and working with wrong version of the programs. We need a development environment which helps us to organize this work.

It is also necessary to have access control. We need to protect the application from unauthorized programmers.

When it comes to data and library allocation, we need to specify these libraries once and for all, in one single place. If we decide to change platform, we need to be sure that we change all physical names at the same time.
If we are going to build applications in a professional way we also need development, test and production levels. It would save a lot of effort and time if we had a development environment that handles program transfer between these levels for us.

Finally, if we could develop and run different applications in the same environment without leaving it, it would be perfect!

Unfortunately, SAS does not provide these utilities, not yet (maybe in a later release). SAS Display Manager is a very nice environment to work in and SAS is a powerful tool, but we have to find our own way to organize our application development.

Niklas Data answer to these needs in SAM.

**SAM - SAS Application Manager**

During all the years we have been working with applications development with SAS we have experienced the need for a development environment so strongly that we decided to build our own. The result is called SAM, the SAS Application Manager.

SAM is a development environment in which you can build your SAS applications. SAM itself is written in SAS/AF®. In order to run it, it is also necessary to have the SAS/SHARE® module. SAS/SHARE handles simultaneous updating of SAS libraries, which is required to make the development environment work.

The first version of SAM was written for the MVS environment on the mainframe platform. We have since then moved it to the PC platform. The current version of SAM has been further developed on PC platform and exists for SAS 6.08, 6.10 and 6.11.

**Utilities in SAM**

- **Program control**
  Program control is handled effectively by SAM. Once accessed by a developer, programs are locked ensuring security. If the administrator wants to know the status of the programs, he or she can browse a control data set at any time, which shows the programs, were they are located and who the owners are.

  SAM handles programs in catalogs, it could be frames, scl, program entries or a catalog entry of any kind. There are no limits in how many catalogs SAM can handle. SAM can also manage program source stored in simple text files.

- **Access control**
  Programs in SAM are protected at supervisor level, only authorized users have access. There are four types of users; Application administrator (access to everything), Developer (can copy from prod, develop and copy to test), Tester (can run programs in test and production levels) and Enduser (can run programs in production level only).
- Automatic allocations
Application libraries are automatically allocated. You just enter the path in a special application data set and SAM takes care of the rest!

- Development, test, production and backup levels
A development-, test-, production- and backup hierarchy is built into the SAM environment. The number of test levels is optional from one up to ten at the most.

All modifications to programs are done in the developer's personal library for that specific application. Only the programs to be modified are copied from the production library. The rest of the application (together with a copy of the former version of the program to be modified) is still in the production library. SAM uses a catalog search path to make this work. To test a new program, the program is moved from the developer's library up to the first test level, tested and then moved up to the next level and so on. The final step, moving the program to production can only be performed by the Application administrator, online or in batch mode. This operation is logged in a SAS table. Every time a program is moved to production, the former version of it is copied automatically to the backup library. The number of backup generations is user defined, up to 99.

When all programs have been moved into production, the test libraries and all development libraries are empty.

- Handle different applications without leaving SAM
The developer can switch between and work with different applications without leaving the environment. SAM reallocates the libraries you need for each application.
SAM is table driven

SAM is controlled by a number of tables. In the following section a few of them are shown.

-The SAMAPPL dataset

In this dataset the application is defined by specifying application name, physical paths to the application libraries and other parameters needed. For each application at least three rows are defined, containing test, production and backup libraries. For each developer one row is added containing the developers personal library, the UTV level. In each UTV row you specify the developers user id, this will give him access to the application. Read-only utility libraries can be specified as General library (GEN). All the SAS applications are defined in the same SAMAPPL dataset.

<table>
<thead>
<tr>
<th>Appl name</th>
<th>Level</th>
<th>Level name</th>
<th>Path</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAAA</td>
<td>BACK</td>
<td>BACK</td>
<td>'g:\sasappl\aaaa\back'</td>
<td></td>
</tr>
<tr>
<td>AAAA</td>
<td>PROD</td>
<td>PROD</td>
<td>'g:\sasappl\aaaa\prod'</td>
<td></td>
</tr>
<tr>
<td>AAAA</td>
<td>TEST</td>
<td>TEST1</td>
<td>'g:\sasappl\aaaa\test1'</td>
<td></td>
</tr>
<tr>
<td>AAAA</td>
<td>TEST</td>
<td>TEST2</td>
<td>'g:\sasappl\aaaa\test2'</td>
<td></td>
</tr>
<tr>
<td>AAAA</td>
<td>TEST</td>
<td>ACCEPT</td>
<td>'g:\sasappl\aaaa\accept'</td>
<td></td>
</tr>
<tr>
<td>AAAA</td>
<td>UTV</td>
<td>DEVELOP</td>
<td>'d:\sasappl\aaaa\dev1'</td>
<td>EMA</td>
</tr>
<tr>
<td>AAAA</td>
<td>UTV</td>
<td>DEVELOP</td>
<td>'d:\sasappl\aaaa\dev2'</td>
<td>ARU</td>
</tr>
<tr>
<td>AAAA</td>
<td>GEN</td>
<td>GENERAL</td>
<td>'d:\general\util'</td>
<td></td>
</tr>
</tbody>
</table>

-The User dataset

In the User dataset the SAM users are defined. The user category is defined in the status variable: ADM (Administrator), UTV (Developer), TEST (Tester) and PROD (Enduser). The language variable holds the SAM language for each user.

<table>
<thead>
<tr>
<th>USER</th>
<th>STATUS</th>
<th>Language</th>
<th>Password</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>EMA</td>
<td>ADM</td>
<td>E</td>
<td>Secret</td>
<td></td>
</tr>
<tr>
<td>ARU</td>
<td>UTV</td>
<td>S</td>
<td>xxxxxx</td>
<td></td>
</tr>
<tr>
<td>SBI</td>
<td>TEST</td>
<td>E</td>
<td>yyyy</td>
<td></td>
</tr>
<tr>
<td>XXX</td>
<td>PROD</td>
<td>E</td>
<td>$sss$</td>
<td></td>
</tr>
</tbody>
</table>

The SAMAPPL and the USER tables together with a few other tables needed to set up an application in SAM are maintained by the Application administrator via a special SAM Administration application.
The Control dataset

The control dataset is used to control the program access and is updated automatically by SAM. The Application administrator can browse this dataset at any time in SAM in order to investigate where a program is located, the date and time when the program was fetched and the name of the user who holds it now. Only programs located in unit or test libraries are shown in the control dataset.

<table>
<thead>
<tr>
<th>PROGRAM</th>
<th>LEVEL</th>
<th>USER</th>
<th>DATETIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAAA</td>
<td>TEST</td>
<td>dev1</td>
<td>95.12.20:10.15</td>
</tr>
<tr>
<td>BBBB</td>
<td>UTV</td>
<td>dev2</td>
<td>95.12.15:08.24</td>
</tr>
</tbody>
</table>

Conclusion

With SAM Niklas Data has a powerful tool to develop programs, not spend time thinking about how to access, share, move, integrate, set in production and maintain programs. SAM supports all these functions and more. For example:

- Program control is handled effectively - once accessed programs are locked ensuring security.
- Applications are protected at supervisor level - only authorised developers have access to applications.
- Application libraries are automatically allocated.
- A development - test - production hierarchy is built into the environment. The developer can switch between and run different applications without leaving the environment.

These functions provide the developer with an environment which significantly improves the chances to successfully develop applications!

We hope that you enjoyed this paper. If you have any questions, please do not hesitate to contact us for further information.

Steve Binning &
Annette Runsteen
Niklas Data AB
P.O. Box 1192
S-172 24 SUNDBYBERG
SWEDEN
+46 8 28 24 24

SAS, SAS/AF and SAS/SHARE software are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries.® indicates USA registration.