APPLICATION DEVELOPMENT UNDER SAS/SHARE®

Peter Dillingham
The Gates Rubber Company

Abstract:

Since SAS/SHARE® does not address catalogs at present, it is necessary to allocate shared, multi-user SAS/AF®
application catalog libraries with a "disp=shr". To quote
Gates' SAS® system consultant Mike Gibson in "Share and
Share Alike" (SUGI 13/1988),

Individual catalog members, therefore, cannot be updated unless all server users
exit the SAS system, and the catalog library
is re-allocated with "disp=old."

This poses a number of problems in managing and maintaining complex, multi-user, shared applications. In Gates' Marketing Communications and Research
Department (MC&R), we have solved the problem of
inaccessible catalogs to some extent by placing all systems
code in a separate partitioned dataset. Application
programs are called from SAS/AF using "%inc
<pdsmembername>" and "xsub <pdsmembername>".
This elevates the need to have all users exit the SAS system
in order to apply fixes or changes to application code. As
an added benefit, we are able to run most data processing
and report generating programs in batch mode, rather than
interactively for better utilization of human and machine
resources.

Introduction:

In Gates' Marketing Communications and Research
Department, we have developed major SAS system
applications for handling purchasing, project management,
advertising inquiry fulfillment, catalog and product literature
inventories customer mailing lists, sales program
administration, and expense budgets and forecasting.

Fifteen departmental users share access to these
applications on a daily basis to generate reports, update
mailing lists, send customized letters and create mailing
labels, track creative projects and expenditures, issue
purchase orders, and fulfill on requests for product literature
and incentive program awards - See Appendix A. Our data
library alone occupies 4000 tracks on our company's IBM
3084 mainframe 3386 disk packs. That we have been able
to computerize all these diverse advertising and
communications functions is due in large part to
SAS/SHARE.

In developing applications for our department, we have tried
to allow users to have as much say as possible in the
creation and modification of the systems they have to work
with on a daily basis. A certain amount of application
tinkering, therefore, goes on almost continually. While
SAS/AF is very powerful and versatile application
development tool, it is difficult to modify in a shared
environment.

Additionally, we have found it much more productive in
terms of people and machine time to run as many of our
application programs in batch mode as possible, rather than
interactively. This means code must reside in a PDS with
appropriate JCL.

Putting Code Where You Can Get At It:

When MC&R application users sign on to the SAS system
under Server, their SAS system Autoexec files "x allocate"
our departmental code library "cic.advert.cntl" using
deptcode as the filerref. Even the simplest of code — such
as invoking a SAS/FSP® Fseedit data entry screen for our
Creative Group's project management system with

    proc fseedit data=dept.traffic
    screen=deptuser.traffic.traffic.screen;
    —

is placed in "deptcode" membername (trsfedit), rather than
in a SAS/AF program. When a user selects the SAS/AF
menu option to edit the dataset traffic, SAS/AF calls the
above code with %inc deptcode (traffic). When the user
is through editing the dataset, he is returned to his main
application menu with a "proc display."

It may seem picayune to place such an insignificant piece
of code in a PDS rather than in a SAS/AF program, but (a)
it is part of the discipline we adhere to in developing
applications for our department, and (b) we have had
occasions where it was necessary to pipe users to a back
up or archived dataset. It is much easier to change "proc
fseedit data=dept.traffic" to "proc fseedit data=dept.traf87"
in a central code library, than sign fifteen users off the
system and have them wait while we access the traffic
application catalog to change a piece of SAS/AF code.

The value of placing shared application code in a
centralized partitioned dataset perhaps becomes more
apparent when dealing with complex data processing
systems. Passing user entered values to a program is one
of SAS/AF's most powerful aspects, but if the bulk of the
program resides in SAS/AF, it suffers the same problem of
inaccessibility in a shared environment, and must be run
interactively. We have gotten around this by using SAS/AF's
"x allocate feature and SAS macro code to push user
entered data values from SAS/AF to our departmental code
library.

To illustrate this, let us look at a concrete application
example. Catalogs and product literature produced by
MC&R are warehoused and fulfilled by an outside supplier.
The processing of customer and salesman orders, however,
and the management of our inventories is done in house
by our staff. If a catalog or product page goes out of stock,
it is put on back order. As back ordered items are reprinted
and put in stock, our order processors enter the item
numbers into the back order system to recall the original
order and generate shipping documents and mailing labels for our contract warehouse. Accessing the back order system option of our Inventory application executes the following code:

```plaintext
X ALLOC (AUTO) DA/CIC.ADVERT.CNTL(AUTOBCK1) SHR REUSE;
PROC DISPLAY C=DEPTUSER.INVENT.AUTOBCK2.PROGRAM; RUN;
```

This allocates a reusable member in deptcode, "Autobck1," with a fileref of "Auto" where user entered data values may be pushed with "===". It also calls the actual fill-in-the-blank AF program "Autobck2.Program" that the order processor will use to enter item numbers returned to stock:

```
ENTER THE AUTOMOTIVE FORM NUMBER(S) THAT ARE NOW IN STOCK:

&N1, _________
&N2, _________
&N3, _________
&N4, _________
&N5, _________

PRESS PF 3/15 TO SUBMIT. TYPE CANCEL TO ABORT.
```

```
== AUTOC
%MACRO SELECT;
##
#N1 NUM(I) EQ "&N1"
#N2 or NUM(I) EQ "&N2"
#N3 or NUM(I) EQ "&N3"
#N4 or NUM(I) EQ "&N4"
#N5 or NUM(I) EQ "&N5"
##
%MEND SELECT;
== X SUBMIT ('CIC.ADVERT.CNTL(SASJCl)') JOBC(BO);
X FREE F(AUTO);
PROC DISPLAY; RUN;
```

Note here the use of the macro "select" to capture the user entered values in the pre-allocated fileref "Auto". If the order processor enters one back order item number, "437-0120" for example, the "===" allocate will pass the following block of code to the PDS member "Autobck1."

```
%MACRO SELECT;
NUM(I) EQ '437-0120'
%MEND SELECT;
```

Also note the "x submit" with its full PDS name and job card statement to execute the actual back order data processing code in batch mode. A portion of our "Autobck0" processing code may be found in Appendix B to show how the macro "select" passed to "Autobck1" is called with a "%include" (Lines 15 & 34).

**Running SAS/AF Applications In Batch Mode:**

As mentioned earlier, placing SAS/AF application code in a separate partitioned dataset also allows us to run most of our data processing and report generating code in batch mode. Many of our applications are extremely dynamic, accessing multiple datasets and carrying out numerous functions such as printing forms and labels, updating data and formats, reducing inventory balances, etc. Running such programs interactively would force our users to spend significant time waiting for the programs to finish. Instead, we have opted to run most of programs in batch mode to free our users for other job related tasks.

As the "Autobck2" program above demonstrates, we are able to accomplish this by using the TSO "X Submit" from within our SAS/AF programs. Mike Gibson our SAS system consultant has provided us with model JCL — see Appendix C — which can be called from SAS/AF as follows,

```
X SUBMIT ('CIC.ADVERT.CNTL(SASJCl)'
' 'CIC.ADVERT.CNTL(MEMNAME)') JOBC(SA);
```

or actually included at the head of a code member in "deptcode". This has given us tremendous flexibility in our application development work, and increased staff productivity.

**Conclusion:**

SAS/SHARE allows multiple users to access SAS/AF applications effectively in a shared environment. Application catalogs, however, cannot be updated unless all SAS system users sign off, and the catalog is re-allocated with a "disp=old". A separate, centralized application code library mitigates this limitation, and allows the developer to run systems code in batch mode for better human and machine resource utilization.

For further information, please contact the author at:

The Gates Rubber Company
Marketing Research and Communications
PO Box 5887
900 South Broadway
Denver, CO 80217
Phone: (303) 744-4200

SAS, SAS/SHARE, SAS/FSP and SAS/AF are registered trademarks of SAS Institute Inc., Cary, NC, USA.

Appendix A:

```
******* MC&R SYSTEMS *******
1. Produce MAILING LABELS.
2. Creative TRAFFIC System.
3. MC&R PURCHASING System.
4. SALES PROGRAM Menu.
5. UPDATE Mailing Lists.
6. Update REPORTS Menu.
7. AD INQUIRY System.
8. INVENTORY System.
9. ANHYDROUS AMMONIA PROJECT.
10. NEW EXPENSE CONTROL SYSTEM.
X. EXIT SAS and TSO
```

---

1543
Datasets Menu

Library: DEPTUSER

<table>
<thead>
<tr>
<th>CMD</th>
<th>MEMBER</th>
<th>MEMTYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>_</td>
<td>ANHY</td>
<td>CAT</td>
</tr>
<tr>
<td>_</td>
<td>COMMON</td>
<td>CAT</td>
</tr>
<tr>
<td>_</td>
<td>EXPENSES</td>
<td>CAT</td>
</tr>
<tr>
<td>_</td>
<td>FORECAST</td>
<td>CAT</td>
</tr>
<tr>
<td>_</td>
<td>INQUIRE</td>
<td>CAT</td>
</tr>
<tr>
<td>_</td>
<td>INVENT</td>
<td>CAT</td>
</tr>
<tr>
<td>_</td>
<td>MAILLIST</td>
<td>CAT</td>
</tr>
<tr>
<td>_</td>
<td>PURCHASE</td>
<td>CAT</td>
</tr>
<tr>
<td>_</td>
<td>REPORTS</td>
<td>CAT</td>
</tr>
<tr>
<td>_</td>
<td>SALEPROG</td>
<td>CAT</td>
</tr>
<tr>
<td>_</td>
<td>TRAFFIC</td>
<td>CAT</td>
</tr>
<tr>
<td>_</td>
<td>UPDATE</td>
<td>CAT</td>
</tr>
</tbody>
</table>

Appendix B:

```sas
** AUTOBCKO — AUTOMOTIVE BACK ORDERS 2/15/88 **/
LIBNAME DEPT SERVER=MCR 'CIC.ADVERTSASLIB';

%INCLUDE DEPTCODE(AUTOBCK1);
DATA DEPT.AUTOSHIP(DROP=1 SW X) WORK.AUTOBCKO;
SET DEPT.AUTOSHIP;
X=TODAYO;
ARRAY NUM(40) NUMBER1-NUMBER40;
ARRAY FLG(40) FLAG1-FLAG40;
ARRAY LIM(40) LIMIT1-LIMIT40;
ARRAY 000(40) ORDQTY1-ORDQTY40;
SW='NO';
DO i=1 TO DIM(NUM);
   IF {
       %SELECT
   } AND FLG(i)='OUT' THEN DO;
       FLG(i)=SUBSTR(PUT(X,MMDDYY6),1,4);
       RELEASE=' ';
   
   
   
Appendix C:

/*SAS EXEC SAS,REGION=2500K,MP=1,RL=1,OUTP=S
SASLIB DD DSN=CIC.ADVERTSASFMT,DISP=SHR
DEPTCODE DD DSN=CIC.ADVERTCNTL,DISP=SHR
DEPTUSER DD DSN=CIC.ADVERTSASPROF,DISP=SHR
LIBNAME DEPT SERVER=MCR 'CIC.ADVERTSASLIB';
*/