Paper 061-2012

How to Use SAS/CONNECT® to Get Your Mainframe to Behave Like a Modern Computer in Your New Business Intelligence Platform

John Hennessey, Social Security Administration

ABSTRACT

This paper reports on the ability to embed SAS/CONNECT® code in a stored process on a SAS® Business Intelligence (BI) platform. The Social Security Administration's Office of Research, Evaluation, and Statistics (ORES) has a BI platform installed on a Windows server to produce summary statistics and data extracts for the analysis of Social Security programs. The BI platform must communicate with large mainframes to access and process the operational data of the Social Security Administration. Since 2006, we have been developing ways to use SAS® Enterprise Guide® and SAS/CONNECT® to bring the mainframe into a modern interface. We are currently developing stored processes that can be used by our internal customers to run directly without ORES intervention.

INTRODUCTION

The Office of Research, Evaluation, and Statistics of the Social Security Administration (SSA) is charged with, among other things, gathering data from our mainframe operational computer systems and doing research and statistical analyses of the Social Security programs. In short, we produce information about the SSA programs. We have recently installed the SAS Business Intelligence Platform on our server to help us modernize our efforts. We have also purchased a copy of SAS/CONNECT for the Platform in order to have it talk to our mainframe that has a traditional installation of SAS, including SAS/CONNECT, and datasets containing hundreds of millions of records. The current financial climate made it inadvisable to request an installation of the BI Platform, or a copy of Integration Technologies, on the mainframe. Therefore, we are testing the ability of SAS/CONNECT installed on the BI Platform on our server to handle the communications between the Platform and the Mainframe. In particular, we are interested in the ability of the new features of the Platform to work with the Mainframe as well as the servers.

STORED PROCESSES

We are currently planning to develop Stored Processes for several existing workloads in ORES. These Stored Processes would be handed off to the users through Web Report Studio or the Add-In to MS Office, whichever they prefer. One of the Stored Processes would need to include work on the mainframe. Therefore, we needed to test the ability to use SAS/CONNECT code within a Stored Process. There were six concerns. Within a Stored Process:

1. Can the SAS/Connect signon code work?
2. Can a MF SAS library be assigned?
3. Can the SAS/CONNECT rsubmit sandwich work?
4. Can a prompt for the MF password work?
5. Can PROCS using a MF dataset be created with the point & click features of EG?
6. Can this Stored Process be used from Web Report Studio and the Add-In to MS Office?

The answer to all of these questions is YES! Here is how it works.

A Stored Process without an Rsubmit Sandwich

1. **Within Enterprise Guide, create a program task that contains the signon script to the MF with your ID and password as macro variables. Below the signon command, create a libname list of all of your SAS libraries on the MF, all commented out. Un-comment the ones that you need in this EG project. (Note the reference to the DB2 database on the mainframe. That must be set up according to your protocols for DB2.)**

```
options comamid=tcp;
%let syxx=syxx.xxx.xxx.gov;
signon syxx.spawner user='&id' password="&password";
*libname abc remote 'xxx.xxxxx.xxx' server=syxx.spawner;
libname def remote 'xxx.xxxxx.yyy' server=syxx.spawner;
*libname ghk remote 'xxx.xxxxx.zzz' server=syxx.spawner;
```

2. **Create prompts for the ID and Password using the EG Prompts Manager Window (View Prompt**

Manager). From the Program Task's Properties/Prompts window, add them to this task.

3. **Run the task. You are now signed onto the MF and your libraries are assigned to your EG session.**
4. **From the Server List window, drag the MF dataset(s) needed into the Process Flow.**
5. **Use the Point & Click capability of EG or create Program windows in EG to perform tasks on the dataset(s), e.g., a frequency of some variables, run a regression analysis, etc.**
6. **Run the entire process flow to make sure it works from beginning to end.**
7. **Point to some blank space in your process flow window and right click. Choose "Create a Stored Process". Follow the instructions for the 5 steps in the Stored Process Wizard as you register the Stored Process. Most of the defaults work. Pay attention to where you are storing the Stored Process.**
8. **Run the Stored Process in EG to make sure it works.**
9. **Test it out in Web Report Studio, Add-In to MS Office, etc. or wherever you wish to use it.**

A Stored Process with an Rsubmit Sandwich

If you want the Stored Process to process one or more of your tasks on the mainframe, then you will need an Rsubmit Sandwich. Therefore, the Top and Bottom Code of the sandwich is needed in those task(s). (See below for examples of Top and Bottom code for the Rsubmit sandwich.) If the task is a code window, simply edit the program to include the Rsubmit sandwich code. If not, it can be done in several ways:

- As you are creating your Stored Process, you are allowed to edit your code. At that point, you can include the Top and Bottom of the Rsubmit Sandwich to the code; or,
- Right click on the Point & Click Task and choose "Add as a code template". This action will create a code window of your point & click task. Modify your process flow to use that code window instead of the Point & Click task icon. Edit that program window to include the Rsubmit sandwich code. I prefer to do it this way so that I can test and modify the process flow before I start the Stored Process registration wizard.

An ODS Sandwich, Please

If you wish to have good-looking output from the mainframe, you can always include an ODS sandwich, even on the MF. The MF doesn't know what it is doing, but it works! Here the trick:

- Create a scratch file on the MF that is fixed block, 80 characters in length.
- Use it in your Top code for the ODS sandwich
- After you close it, Proc Download it to your BI Platform
- Even though it does not open in Enterprise Guide at the end of running your Stored Process, it gets the job done! You can now go and get it from wherever you sent it.
- Remember, the BI Platform is controlling everything, so you must describe the path for the download from the point of view of the BI Platform, not your PC.

Rsubmit Top Code Example

Below is an example of the top code for an Rsubmit sandwich to the MF. A few comments:

- The macros are ones that are generated and used with EG and so must be created on the MF.
- For some ODS destinations, the option, trantab = ASCII, must be included since the MF is EBCDC.
- Even though the libraries have been assigned before, the assignment was local for EG to know about the libraries. The MF must be told about these libraries when you use the Rsubmit sandwich, but, this time, they are local, so, no remote engine.

```
rsubmit syxx.spawner;
 %macro _SASTASK_DROPDS(dsname);
      %if %sysfunc(exist(&dsname)) %then %do;
            proc sql;
                  drop table &dsname;
            quit;
       %end;
       %if %sysfunc(exist(&dsname,view)) %then %do;
            proc sql;
                  drop view &dsname;
            quit;
        %end;
```

```
%mend _SASTASK_DROPDS;
%macro _eg_conditional_dropds(dsname);
      %if %sysfunc(exist(&dsname)) %then %do;
            proc sql;
                  drop table &dsname;
            quit;
       %end;
       %if %sysfunc(exist(&dsname,view)) %then %do;
            proc sql;
                  drop view &dsname;
            quit;
        %end;
%mend _eg_conditional_dropds;
ods rtf file='xxx.xxxx.tran' trantab=ascii style=sasweb;

*libname tbi_comb 'e:\xxx\xxx\xxx';
libname earn 'xxx.xxxx.xxxxxxxx';
*libname bhg 'xxx.xxxxx.xxxx';
```

Rsubmit Bottom Code Example

Below is an example of the bottom code for the Rsubmit sandwich. The binary option must be used in order for the MF not to change anything.

```
ods rtf close;
proc download infile='xxx.xxxxx.tran'
      outfile='m:\xxx\xxx\output.rtf' binary;
run;
endrsubmit;
```

CONCLUSION

Many businesses still rely on powerful mainframe computers to run their real time operations. The MF also stores large quantities of operational data that need to be converted to informational data for analysis. In today's world, it is hard to keep up with the amount of data about the business that needs to be analyzed. Therefore, we need to make use of every powerful processing capability that exists within the business.

Our results suggest that an installation of SAS/CONNECT on the SAS BI Platform will allow one to bring the powerful capability of the Mainframe into the BI workload to preprocess large amounts of data before it is brought down to the BI Platform itself for further processing and analysis. In addition, this MF processing can be done in parallel to the work done by the BI Platform. This approach avoids the expensive purchase of the SAS BI Platform software for the MF and allows a traditional installation of SAS on the MF to function within the workload of the BI Platform, i.e. the MF will behave like a modern computer within your SAS BI Platform.

REFERENCES

1. John Hennessey, "SAS® Enterprise Guide® and SAS/CONNECT® Software –Peaceful Co-Existence Using SAS Code", SUGI 31
2. John Hennessey, "SAS/CONNECT® to Mainframes and Servers Within Enterprise Guide® Version 4 Using SAS Code", SGF 2007
3. John Hennessey, "Using SAS® Enterprise Guide® and SAS/CONNECT® to Make the Mainframe Behave Like a Modern Computer", SGF 2008
4. John Hennessey, "A SAS® Enterprise Guide® 4.2 and SAS/CONNECT® Mainframe Project which Makes the Mainframe Behave Like a Modern Computer", SGF 2010

CONTACT INFORMATION

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

John Hennessey
Office of Research, Evaluation, and Statistics
Social Security Administration
6401 Security Blvd.
4-C-15 Operations Bldg.
Woodlawn, MD 21235
Work Phone: (410) 965-0102
E-Mail: John.C.Hennessey@ssa.gov