

SAS® as a Multi-Purpose Tool in the Healthcare Environment

Robert Muzny, INTEGRIS Health, Oklahoma City, OK

ABSTRACT

The healthcare industry offers many opportunities to learn about the countless uses of the SAS System. With the varied number of information systems that are used to maintain patient care data, personnel data, and business-decision data, we quickly learn there is a need for a tool that can work between these systems. Business decision makers need their unique pieces of the puzzle combined with others to form a comprehensive view of the picture.

INTRODUCTION

The healthcare industry maintains many worlds where data exists. Some of these worlds are friendly while others are not. At times, a programmer may have requests to meld these differing worlds. This is the joy of a SAS programmer. Other times, we are simply called upon to develop reports that vendor products should be able to produce, but fail. Finally, we are also called upon to automate tasks that can be accomplished through manual means, but the tedious boredom of the manual task is unnecessary and wasteful of personnel resources.

This paper was written to emphasize a few of the opportunities that the Decision Support department has had to please its internal customers. We have been able to reduce our own workload by automating many manual tasks. We have improved upon vendor products by providing additional services to business users. Through the SAS EIS product, we have opened a door to the data so that decision makers can formulate their own questions, and find their own answers.

AUTOMATE DAILY PROCESSING OF RAW SOURCE FILES INTO USEFUL INFORMATION

Batch processing is a handy method used to update daily information. Data files can be accessed during off hours, and various types of output can be updated for business users. We currently use SAS to provide daily HTML graphical pictures of how the business is doing; update Microsoft Access databases for staff; update MDDb's that are contained within Executive Information Systems; as well as send reports to e-mail addresses and various printers around the campus.

Tasks are scheduled on a Windows 2000 server. These tasks are standard ".BAT" files that call SAS routines. The .BAT file in example 3 includes the SAS executable along with the configuration file that is to be used (-CONFIG) and the SAS program that is to be executed (-SYSIN). Each SAS program has the necessary LIBNAME statements included within it to fully access and store data. It also includes the output destinations for data and ends with an ENDSAS command.

Example 1: BAT file

```
"x:\Program Files\SAS\V8\sas.exe" -CONFIG "z:\V8\sasv8Dev8.cfg" -SYSIN
z:\develop_eis\eis_data\MonthlyPrograms\Cerner\DailyCCLHarry.sas

"x:\Program Files\SAS\V8\sas.exe" -CONFIG "z:\V8\sasv8Dev8.cfg" -SYSIN
z:\develop_eis\eis_data\NightlyJobs\DailyEISUsage.sas

"x:\Program Files\SAS\V8\sas.exe" -CONFIG "z:\V8\sasv8Dev8.cfg" -SYSIN
z:\develop_eis\eis_data\MonthlyPrograms\Cerner\DailyCCLImport_SMC.sas
```

SAS programs that are run in this manner automatically have LOG files output to the directory where the .BAT file was called. This allows the programmer to review what occurred during the session. Errors within the program can be trapped programmatically and emails can be sent to the programmer's email address or pager number.

The following information on bat files was gleaned from the SAS-L discussion list, and was authored by Mark Terjeson, with the Department of Social and Health Services in the state of Washington:

If you are looking for command line syntax for invoking SAS then here are a couple of examples:

To invoke sas and tell it what code to run:
sas.exe -sysin myprogram.sas

Of course you may need to add drive and paths:
c:\yoursaspath\sas.exe -sysin d:\mydir\myprogram.sas

If you want to make sure what autoexec.sas get used:
c:\yoursaspath\sas.exe
-autoexec d:\sas\autoexec.sas
-sysin d:\mydir\myprogram.sas

<of course commands are all on one line, I am breaking across lines here just because postings usually wrap kinda ugly>

If you want to make sure what config.sas get used:
c:\yoursaspath\sas.exe
-autoexec d:\sas\autoexec.sas
-config d:\sas\config.sas
-sysin d:\mydir\myprogram.sas

If you want to make sure where the log goes:
c:\yoursaspath\sas.exe
-autoexec d:\sas\autoexec.sas
-config d:\sas\config.sas
-log d:\work\logs\pull_sample200102131244.log
-sysin d:\mydir\myprogram.sas

If you want to turn off the logo banner from displaying on screen:
c:\yoursaspath\sas.exe
-autoexec d:\sas\autoexec.sas
-config d:\sas\config.sas
-nosplash
-log d:\work\logs\pull_sample200102131244.log
-sysin d:\mydir\myprogram.sas

other available options:

- xwait
- print <file> (to redirect your print output window)
- nodms (turns off the display manager)
- nologo (turns off the logo)(for earlier versions, replaced by nosplash)
- nosplash (to suppress the momentary flash of the SAS logo)

SAS PROGRAMS THAT READ DATA AND CREATE OTHER SAS PROGRAMS FOR GREATER PRODUCTIVITY

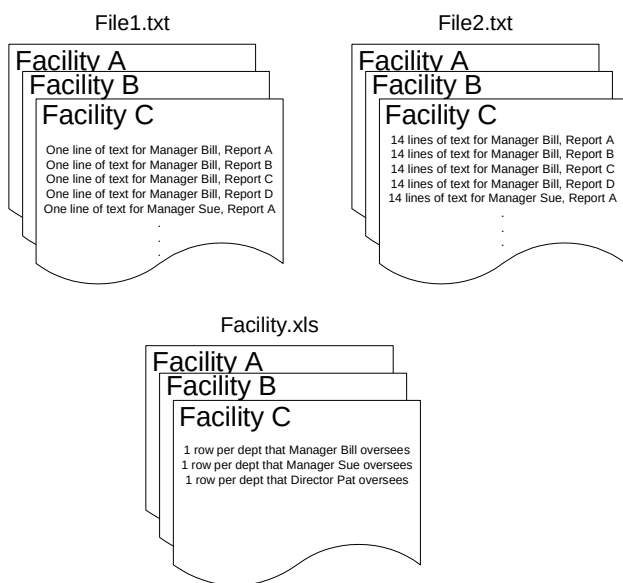
One of the joys of working with SAS is the opportunity to up-end a vendors bulky process. This is especially joyful when you've been able to reduce the workload of a co-worker. One case in particular deals with a productivity system that included a very unproductive bi-weekly processing task list.

Consider this scenario:

You are tasked with producing productivity reports for 2 metro facilities and 8 smaller rural facilities on a bi-weekly basis. This is accomplished, according to the vendor, by maintaining two text files that outline the reports that are to be produced and the parameters that the software program needs to correctly produce the files. These two text files

must be in a very specific format, and must correlate with each other if the process is to complete successfully.

In addition, an Excel spreadsheet is maintained for the productivity system. This spreadsheet lists the managers within the organization, and which business centers they manage. If a manager manages more than one business center, then they also are assigned a "Roll-Up Group" or "RUG". This spreadsheet is imported into the productivity system so that the software can link a manager to a business center.



Further, after the reports have been created, you will need to E-mail each manager INDIVIDUALLY to let them know where their unique reports are stored.

This process is run every two weeks. It is run for two metro facilities (just under 1200 reports), and eight rural facilities, with a total count of just over 1300 reports. A little math: 2500 reports every two weeks. The accumulation of lines in file1.txt is around 2500 lines. File2.txt is around $2500 * 14 = 35,000$ lines. Facility.xls is simpler with around 500 lines.

Each time a new manager comes on board, or a promotion occurs, or someone leaves a facility, then the Excel spreadsheet changes and the two TXT files must change also. The E-mail list changes as well. All of this is to be done manually, *according to the vendor*.

The SAS solution to this is to key in on the Excel spreadsheet. This single source of information is easy to maintain, and can be read in for the development of all the other pieces. SAS will read the Excel file, perform various activities that allow us to identify which manager needs what type of report, and then output the two TXT files (in sync!) that the vendor program needs. After the vendor program does its work and has output the data files to their correct location (based on our SAS program), then we run another SAS program that E-mails the managers and displays text unique to that manager's specific report location.

The e-mail portion is accomplished by reading the Excel file (which includes the e-mail address of all managers), and then we dynamically write out a text file and label it with a .SAS extension. The .SAS file is created using PUT statements, as seen in Example 2. Output is in Example 3.

Example 2: PUT statements that create the .SAS file

```
DATA _null_;
  SET e_;
  BY DESCENDING a_email rug;
  FILE "\\N2BMD09\SASPgms\Email.sas" NOTITLE NOPRINT LRECL=300;

  PUT @1 "FILENAME mail EMAIL TO="
    +0 trimmail
    +(-1) "@integris-health.com" ;
  PUT @1 " SUBJECT='&facility Productivity Reports for Period Ending "
```

```

+0 pdate +0 "';";

PUT @1 "DATA _NULL_";
PUT @5 "FILE mail;";
PUT @1 "PUT '&pperiod';" ;
PUT @1 "PUT ' '";
PUT @1 "PUT ' '";
PUT @1 "PUT 'Your current period productivity reports can be accessed by
clicking the link below. '";
PUT @1 "PUT ' '";
PUT @1 "PUT 'If you have a variance greater than 1 FTE, you will need to submit
a Variance Report. Please see the Prod Detail Var report for the variance
calculation. '";
PUT @1 "PUT ' '";
PUT @1 "PUT 'Click here >>> \\N2BMD07\VW-REPORTS\$\" +0 trimmail +0 "';";
PUT @1 "PUT ' '";
PUT @1 "PUT 'For information regarding the layout of the reports, go to
http://inside/ihdecisionsupport/Visionware_Defined2.doc '";
PUT @1 "RUN;";
PUT @1 "*-----";

RUN;

```

Example 3: Final Email.SAS file

```

FILENAME MAIL EMAIL TO='W00TAA@integris-health.com'
SUBJECT='Metro Productivity Reports for Period Ending 12/13/2003 ';
DATA _NULL_;
FILE mail;
PUT '*** Payroll Period: 11/30/2003 - 12/13/2003 ***';
PUT ' ';
PUT ' ';
PUT 'Your current period productivity reports can be accessed by clicking the link
below. ';
PUT ' ';
PUT 'If you have greater than 1 FTE variance, you will need to submit a Variance
Report. Please see the Prod Detail Var report for the variance calculation. ';
PUT ' ';
PUT 'Click here >>> \\ N2BMD07\VW-REPORTS\$W00TAA ';
PUT ' ';
PUT 'For information regarding the layout of the reports, go to
http://inside/ihdecisionsupport/Visionware_Defined2.doc ';
RUN;
*-----;
FILENAME MAIL EMAIL TO='W00DBA@integris-health.com'
SUBJECT='Metro Productivity Reports for Period Ending 12/13/2003 ';

DATA _NULL_;
FILE mail;
PUT '*** Payroll Period: 11/30/2003 - 12/13/2003 ***';
PUT ' ';
PUT ' ';
PUT 'Your current period Productivity reports can be accessed by clicking the link
below. ';
PUT ' ';
PUT 'If you have a greater than 1 FTE variance, you will need to submit a Variance
Report. Please see the Prod Detail Var report for the variance calculation. ';
PUT ' ';
PUT 'Click here >>> \\ N2BMD07\VW-REPORTS\$W00DBA ';
PUT ' ';
PUT 'For information regarding the layout of the reports, go to
http://inside/ihdecisionsupport/Visionware_Defined2.doc ';
RUN;
*-----;
.
.

```

When we execute the Email.SAS program, all of the emails are sent out one at a time to the designated recipient. Each email has a different message (where the reports are located). The only "distribution list" that is kept is the original Excel spreadsheet that tells us which managers are in charge of which departments.

As a side note, after developing the process above, in which we write a SAS program to create another SAS program, the author learned more about macros. The code could be modified and "macro-tized", but I believe the code as it now sits is easy to maintain and will be understood by future staff members.

CONVERT DATA FROM "STANDARD" REPORTS INTO WHAT THE REQUESTOR ACTUALLY WANTS

"Buy it! Buy it!" managers say, primarily due to tremendous sales teams that show what their software will do. "Canned reports are already available!"... "No programming knowledge necessary!" they say. For the majority of information needs, this *might* be true. It has been my experience that there are *always* requests for information that do not fit into the canned reports. This is not said to fault the developers of the software product, it is simply a fact: we are a curious mass of humanity. As we look at data and identify the information within it, we arrive at more questions. These questions eventually move beyond the economic limit of developing canned reports.

When canned reports no longer meet the requestor's needs, then we look to the software package for other output means. Does the tool provide ad-hoc reporting capabilities? Can ad-hoc reports be set up as scheduled tasks? Can they be saved to a network location or perhaps printed for the requestor? What is the price per report? This is when we call in the Big Dog: SAS

My employer has a patient charge system that was negotiated with a limit to the number of reports that are produced. When the facility reaches the limit, then each additional report comes with a fee. A monthly report incurs 12 "report fees" per year. A bi-weekly report incurs 26 report fees. The same charge structure applies for weekly and daily reports. Added to the financial burden is the fact that our data is maintained for us, at a remote site where our queries cannot touch the data. The only method our local staff has is pulling soft-copy reports

SAS is used in this environment to read various soft-copy reports, merge the data based on common keys (facility, department, patient ID), and then produce output in the form of a standard text file, Excel, or HTML.

SAS programming features that are useful when reading soft-copy reports:

INPUT @XX NAME @;

Use of the trailing "@" will hold the record pointer at the current line, rather than moving on to the next line for input. There are many instances when it is helpful to input a portion of the current line to identify where we are in the report. We then want to select a different portion of that line, or move on to the next line.

Example 1: Trailing @

```
DATA foo;
    RETAIN RptDate RptPage Facility; * RETAIN will keep our header;
                                     * information as we read the detail records;
    FORMAT postdate mmdyy10.;
    INFILE "Z:\InputFiles\SoftReportA.txt" MISSOVER LRECL=126 END=eof;

    INPUT @1 NewLine $126. @;          * Note use of trailing @ to hold the line pointer;

    IF SUBSTR(NewLine,27,3) = "BMC"
    THEN DO;
        *** This tells us we have a header row;
        INPUT @1 RptDate mmdyy10. /* Report Date */
               @42 RptPage 4. /* Report Page */
               @27 Facility $3. /* Facility */;
    END;
    ELSE IF SUBSTR(NewLine,52,1) in ('1','2','3','4','5','6','7','8','9','0')
        AND SUBSTR(NewLine,56,1) in('1','2','3','4','5','6','7','8','9','0')
    THEN DO;
        *** This tells us we have a detail row;
        INPUT @24 costctr $3. /* Cost center number */;
```

```

@42    pcu          7.          /* PCU number */
@19    pcuqty       4.          /* PCU quantity */
@105   pcuamt       10.2        /* PCU amount */
@117   postdate     mmdyy10.    ;

ELSE INPUT;  * No place holder here, continue on to the next line;
RUN;

```

FILENAME FILEREF('FILE LOCATION 1', 'FILE LOCATION 2', 'FILE LOCATION 3');

The filename statement enables the programmer to specify a range of incoming data sets that will be used in the DATA step. Within the DATA step use INFILE along with the fileref to read the specified files.

Example 2: Filename

```

FILENAME readme('\\N2bmd98\Visionware\Payroll\CCREATIF\ccreatif*.*'
, '\\N2bmd98\Visionware\Payroll\CCREATIF_History\ccreatif101803'
, '\\N2bmd98\Visionware\Payroll\CCREATIF_History\ccreatif110103'
, '\\N2bmd98\Visionware\Payroll\CCREATIF_History\ccreatif111503'
, '\\N2bmd98\Visionware\Payroll\CCREATIF_History\ccreatif112903');

DATA a(KEEP=ssn emplname depttype srvcdate paytype payhrs testdept paydept);
INFILE readme          DELIMETER='$' MISSOVER TRUNCOVER DSD;
INPUT  . . .
.
.
.
RUN;

```

RETAIN;

The RETAIN statement asks SAS to preserve the value assigned to a variable from the current iteration of the DATA step to the next. Without the RETAIN statement, the SAS System automatically sets variables to missing before each iteration of the data step. When reading a report, it is often helpful to identify some header information, and then RETAIN that information through each successive iteration of the INPUT statement that reads detail records.

An additional feature of the RETAIN statement is the ordering of your output variables. If the RETAIN statement is placed before the SET statement, then the listed variables will appear at the front of the output SAS data set, in the order listed in the RETAIN statement.

Example 3: Retain

```

DATA foo;
  RETAIN LoLOS HiLOS; *RETAIN will preserve the value of these variables;
                        *through the next iteration;
  SET bar;
  LoLOS=min(LoLOS,LOS);    * Find the minimum LOS value ;
  HiLOS=max(HiLOS,LOS) ;   * Find the maximum LOS value;
RUN ;

```

SUBSTR AND SCAN

Two functions that stand out when reading soft-copy reports are SUBSTR (substring) and SCAN. SUBSTR allows the developer to extract a section of text from a variable by listing the starting position and the number of characters to extract. SCAN looks for a particular character within the text, and then returns the text string that precedes that character. I often use it when breaking apart codes of varying lengths.

Example 4: SUBSTR and SCAN

```

DATA testdata;
  Initial_value = '11236-5-ABC54'; OUTPUT;
  Initial_value = '1136-45-AB54';  OUTPUT;
  Initial_value = '113-5';          OUTPUT;
RUN;

```

* Break the value into 3 pieces, we also want to know what is in positions 1, 2, and 3;

```

DATA testdata_modified;
  FORMAT Value_part_a Value_part_b Value_part_c $5.;
  SET testdata;
  Value_part_a = SCAN(initial_value,1,'-');
  Value_part_b = SCAN(initial_value,2,'-');
  Value_part_c = SCAN(initial_value,3,'-');
  Pos123      = SUBSTR(initial_value,1,3);
RUN;

```

The results of this data step are:

Initial Value	Value Part A	Value Part B	Value Part C	Pos123
11236-5-ABC54	11236	5	ABC54	112
1136-45-AB54	1136	45	AB54	113
113-5	113	5		113

CONCLUSION

You are already a SAS programmer, else you would not be at this conference (nor would you be pouring through the proceedings!) This paper was written to emphasize a few of the SAS tricks that I have found useful. Similar to assisting my soccer teammates to slide a backdoor goal in past the defenders, I have had many joyous moments when a co-worker accomplishes a task that has been made possible with a bit of programming ingenuity.

When reviewing a data conversion or reporting task, I find that SAS will typically be the tool of choice for quick turn-around and long-time repetition. I have often found myself in the scenario where I have a hammer, and everything looks like a nail. However, by learning various tricks that SAS has, you may find that the hammer in your grip can quickly morph into a screw-driver, wrench, or a power tool.

ACKNOWLEDGEMENTS

I would like to thank the staff of Decision Support for welcoming suggestions and trusting that the code being written will enhance their final products. Without them, and a Director that is always looking over the horizon for the next project, this paper would not have been written. Thanks also to the many contributors on SAS-L. If you do not know what SAS-L is, open your browser to <http://www.listserv.uga.edu/archives/sas-l.html>

CONTACT INFORMATION

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

Robert Muzny
 INTEGRIS Health
 5300 North Independence, Suite 230
 Oklahoma City, OK 73112
 Work Phone: 405.951.2121
 Fax: 405.945.5408
 Email: Robert.muzny@integrish-health.com
 Web: www.integrish-health.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.