Paper 143-25

Using a SAS/IntrNet™ Data Browsing Application to Generate SAS Code for Stand-alone Reports

Paul Gilbert, DataCeutics Inc.

ABSTRACT

At DataCeutics, Inc., we have developed a SAS/IntrNet-based tool for browsing SAS data and generating listings, summary tables and figures that assist in reviewing clinical trial data. This tool's primary user base is the non-technical clinical scientist. However, we saw an opportunity for the tool's use to be extended to technical users such as statisticians and clinical programmers. To do this we needed the ability to generate the underlying SAS code that created the output and then save the code to a file where the statisticians and clinical programmers could further modify it and execute it in a batch environment. For this to work, we developed a self-contained code generation 'button' that could be easily added to the dynamic SAS/IntrNet-generated HTML report generation pages.

This paper describes the application, how the code generation functionality was designed and how the code generation button was implemented.

INTRODUCTION

In this discussion we take a top-down approach to describe the design of the code generation function of the application. In this paper we will discuss:

- SAS data browsing and report generation;
- Listing generation tool;
- Code generation button;
- SAS code used to create the code generator.

The last section contains an example of code that was generated.

BROWSING AND REPORT GENERATION APPLICATION

The application home page, see Figure 1, displays drugs that are available for browsing.

- User selects a drug; all studies for the selected drug are then displayed.
- User selects a study; all data files for the selected study are then displayed.
- User selects a data file, all reporting functions for the selected data are then displayed on a toolbar.
- At this point, see Figure 2, the user can select from the functions: Display Columns, Subset Data, Data Review, Generate Reports, Interactive Graph or Patient Review.

Note that each individual study is registered to the application. The registration process consists of making an entry into a SAS dataset, describing the location of the data files and format library. Figure 2 is a display of selections for Drug 1, Study 001 and the Demographics data.

The code for this application is stored in a directory on the SAS/IntrNet server in the form of several SAS programs.

FIGURE 1- APPLICATION HOME PAGE

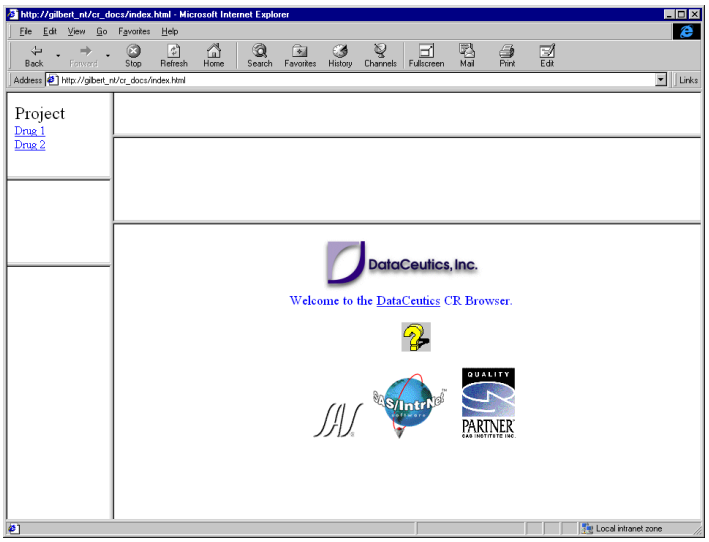
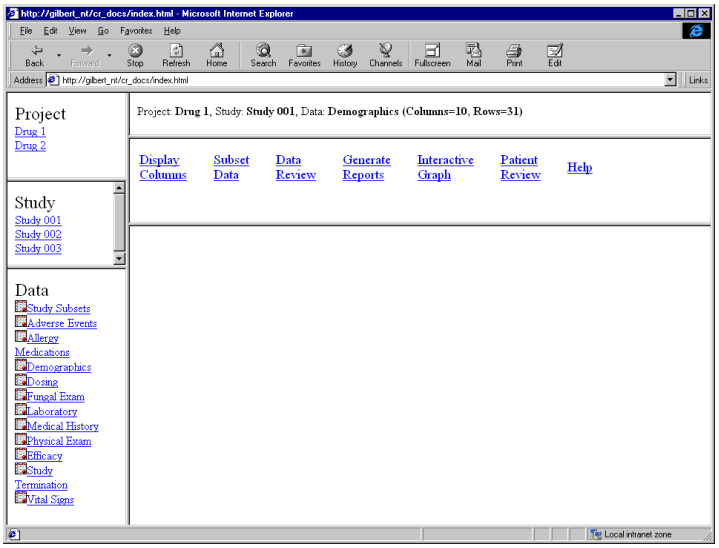


FIGURE 2 - SELECTION FOR DRUG1, STUDY 001 AND DEMOGRAPHICS DATA



LISTING GENERATION TOOL

The Generate Listing tool contains two pages:

- The first page, Figure 3, contains a selection list of variables to display.
- The second page, Figure 4, contains selections ordering data, a line break, titles, footnotes, output orientation, download to Excel and generate code.

The Generate Listing tool was designed as a two-page process. The first page is used to gather the appropriate variables; they are then used to create selection list on the second page. Figure 5 is a display of the listing that was generated by the selections.

CODE GENERATION BUTTON

This tool's target user base was originally non-technical clinical scientist. However, the tool's use can be extended to technical users such as statisticians and clinical programmers. To add this capability we needed the ability to generate the underlying SAS code that created the output and then save the code to a file where the statisticians and clinical programmers could further modify it and execute it in a batch environment. For this to work, we needed to develop a self-contained code-generating 'button' that could be easily added to the dynamic SAS/IntrNet-generated HTML pages.

The code generation function was designed as a SAS macro. To create the code generation button, the following code as added to the program that creates the second selection page (Figure 4). This code creates an HTML selection box named "code". The value of "code" is passed to the next HTML page as the macro variable **&code**, with a value of Y or N.

```
'Generate Code<br>'/
'<SELECT NAME="code">'/
  '<OPTION VALUE="N">No'/
  '<OPTION VALUE="Y">Yes' /
'</SELECT><hr>'/
```

To implement the code generation, the following code was added to the final program of the Generate Listing tool. The second selection page (Figure 4) calls the final program.

```
%gencode(gen=&code, outname=&outname);
```

The macro variable **&code** is then used by the **%gencode** macro call located in the last listing generation program . This simple two step design allows us to implement the code generation on many reports in a timely manner.

FIGURE 3 LISTING SELECTION PAGE 1

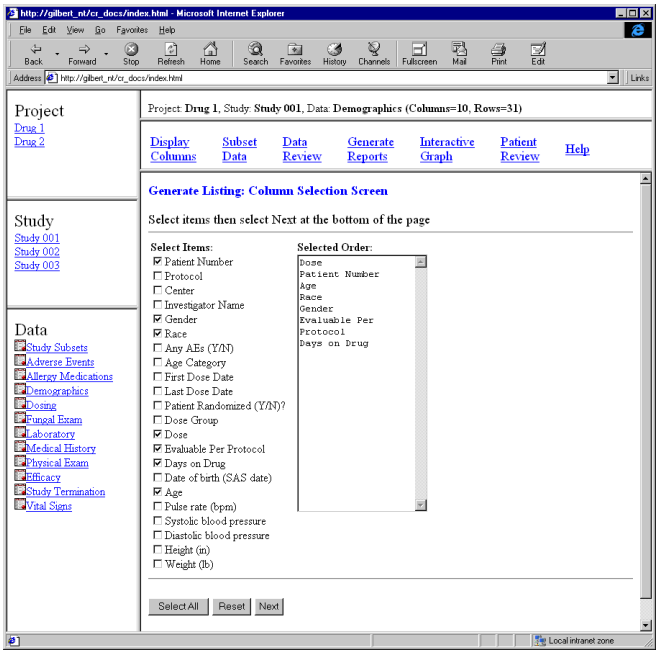


FIGURE 4 - LISTING SELECTION PAGE 2

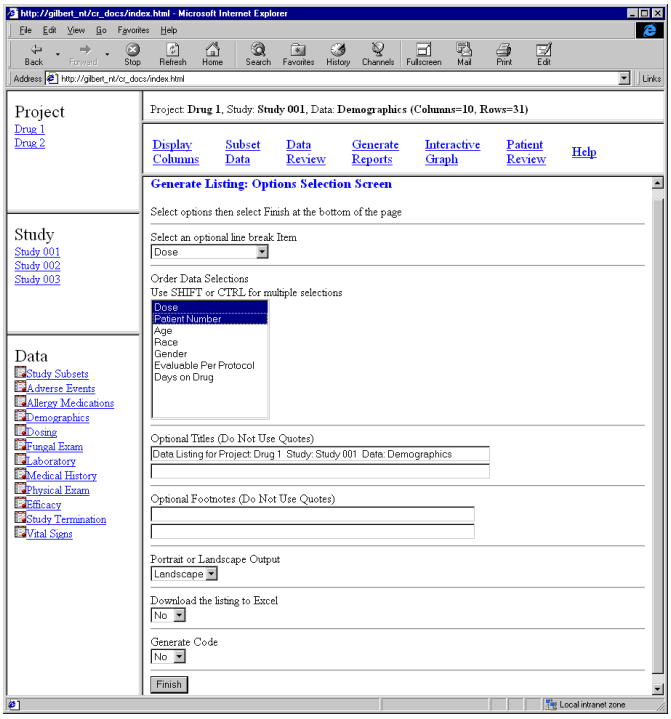
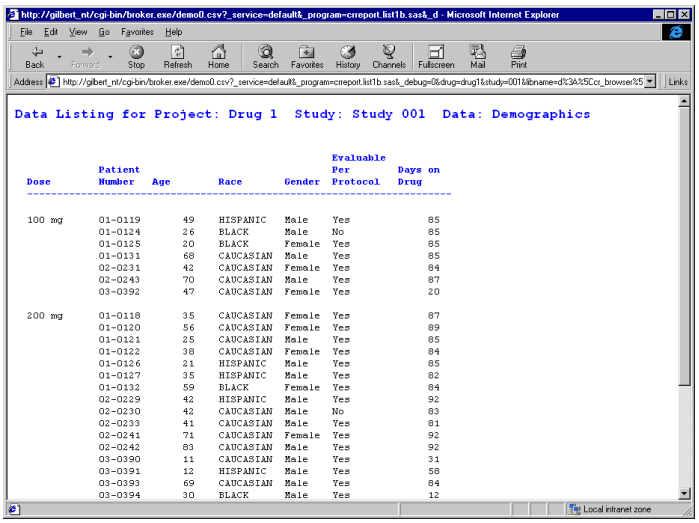


FIGURE 5 - LISTING RESULTS PAGE



Data Listing for Project: Drug 1 Study: Study 001 Data: Demographics

Dose	Patient Number	Age	Race	Gender	Evaluable Per Protocol	Days on Drug
100 mg	01-0119	49	HISPANIC	Male	Yes	85
	01-0124	26	BLACK	Male	No	85
	01-0125	20	BLACK	Female	Yes	85
	01-0131	68	CAUCASIAN	Male	Yes	85
	02-0231	42	CAUCASIAN	Female	Yes	84
	02-0243	70	CAUCASIAN	Male	Yes	87
	03-0392	47	CAUCASIAN	Female	Yes	20
200 mg	01-0118	35	CAUCASIAN	Female	Yes	87
	01-0120	56	CAUCASIAN	Female	Yes	89
	01-0121	25	CAUCASIAN	Male	Yes	85
	01-0122	38	CAUCASIAN	Female	Yes	84
	01-0126	21	HISPANIC	Male	Yes	85
	01-0127	35	HISPANIC	Male	Yes	82
	01-0132	59	BLACK	Female	Yes	84
	02-0229	42	HISPANIC	Male	Yes	92
	02-0230	42	CAUCASIAN	Male	No	83
	02-0233	41	CAUCASIAN	Male	Yes	81
	02-0241	71	CAUCASIAN	Female	Yes	92
	02-0242	83	CAUCASIAN	Male	Yes	92
	03-0390	11	CAUCASIAN	Male	Yes	31
	03-0391	12	HISPANIC	Male	Yes	58
	03-0393	69	CAUCASIAN	Male	Yes	84
	03-0394	30	BLACK	Male	Yes	12

CODE GENERATION FUNCTION

The key to generating the code was to develop a SAS macro to accomplish this task. The macro, listed below consists of three sections.

- The first section collects the current state, application and page. The term "current state" refers to all of the application settings and page selection settings that are defined for the current report being executed. The SAS/IntrNet software and this application are designed to pass and store all selections as SAS macro variables. The current state of the application and page was accomplished by selecting all of the current macro variable settings and writing them to a file. These macro variables and values reside in the SAS dictionary.macros table. See ***Step 1:*** in Figures 6 & 7.
- The second section appends the SAS code that is currently being executed, to the file created above. This is accomplished using a copy command. See ***Step 2:*** in Figures 6 & 7.
- The third section is to write back an HTML page confirming the Code generation. See ***Step 3:*** in Figure 6.

The **%gencode** macro is displayed in figure 6. The code generated from the selections in figure 4 is displayed in figure 7.

FIGURE 6 - %GENCODE MACRO

```
%macro gencode(gen=N, outname=);

%if &gen=Y %then %do;
Step 1:
  proc sql;
  create table _macro as
  select * from dictionary.macros
  where scope ne 'AUTOMATIC'
  order by scope, name;

  data _macro; set _macro end=eof;
   if name in ('_ADMAIL','_ADMIN','_DEBUG','_HTCOOK',
         '_HTMOVP','_HTMTITL','_HTREFER','_HTUA'
         '_RMTADDR','_RMTHOST','_RMTUSER',
         '_SERVICE','_URL','_VERSION') then delete;
   if name='CODE' then value='N';
   if name='GEN' then value='X';
  run;

  filename _test "&_TMPDIR.&outname";
  data _null_; set _macro end=eof;
    file _test;
    if _n_=1 then put
    "*****************************************************************"/
    "* Code Generated by CR Browser"/
    "* &sysdate &systime"/
    "* &title1 "/
    "*****************************************************************";"//
    "options sasautos=('d:\cr_browser\macros',"/
    ",SASAUTOS);"/;

    put '%let ' name +(-1) '=' value +(-1) ';';

    if eof then put ' '//;
  run;

Step 2:
  options noxwait;
  data _null_;
    command="copy
&_TMPDIR.&outname+d:\cr_browser\reports\&_pgm..sas
&_TMPDIR.&outname";
    call system(command);
  run;

Step 3:
  data _null;
   file _webout;
   put
   'Content-type: text/html' //
   '<HTML>' /
   '<BODY BGCOLOR="WHITE" LINK="green" VLINK="green"
ALINK="red">' /
   '<FORM ACTION="/cgi-bin/broker.exe/"
TARGET="sasoutput" ALIGN="right">' /
   '<INPUT TYPE="hidden" NAME="_service"
VALUE="default">' /
   '<INPUT TYPE="hidden" NAME="_debug" VALUE="0">' /
   '<FONT SIZE=+1 COLOR=blue>'/
   "<b>Code is Generated in &_TMPDIR.&outname</b><br>"/
   '</FONT>'/
   '</FORM>' /
   '</BODY>' /
   '</HTML>';

%end;
%mend;
```

FIGURE 7 - CODE GENERATED

```
*****************************************************************
* Code Generated by CR Browser
* 01OCT99 10:09
* Data Listing for Project: Drug 1  Study: Study 001  Data:
Demographics
*****************************************************************;

options sasautos=('d:\cr_browser\macros',
      SASAUTOS);
Step 1:
%let GEN=Y;
%let SQLOBS=0;
%let SQLOOPS=0;
%let SQLRC=0;
%let CODE=N;
%let DATASET=demo0;
%let DRUG=drug1;
%let FOOT1= ;
```

```
%let FOOT2= ;
%let FORMATS=d:\cr_browser\data\drug1\format;
%let IOUTPUT=L;
%let LENG1=10;
%let LENG2=7;
%let LENG3=8;
%let LENG4=2;
%let LENG5=2;
%let LENG6=1;
%let LENG7=8;
%let LIBNAME=d:\cr_browser\data\drug1\001;
%let OUTNAME=temp.sas;
%let PATID=sub;
%let SKIP=DOSE_FMT;
%let STUDY=001;
%let TITLE1=Data Listing for Project: Drug 1  Study: Study 001
Data: Demographics;
%let TITLE2= ;
%let VAR=DOSE_FMT;
%let VAR0=2;
%let VAR1=DOSE_FMT;
%let VAR2=SUB;
%let VARCNT=7;
%let VARNM1=DOSE_FMT;
%let VARNM2=SUB;
%let VARNM3=AGE;
%let VARNM4=RACE;
%let VARNM5=SEX;
%let VARNM6=EVAL;
%let VARNM7=DOSEDAYS;
%let WC= ;
%let _PGM=list1b;
%let _PGMCAT= ;
%let _PGMLIB=crreport;
%let _PGMTYPE=sas;
%let _PROGRAM=crreport.list1b.sas;
%let _TMPDIR=D:\temp\;
```

Step 2:

```
*************************************************************************
**    Program:    list1b.sas
**    Author:     DataCeutics, Inc.
**    Language/Version: SAS/6.12 TS045
**    CR Browser Version: 1.0.0
**    Copyright DataCeutics, Inc 1999
**    Description: Runs the %print1 for the listing.
*************************************************************************;
%macro runit;
  **Listing generation code removed to save space**.
%mend;

%setup(in=&libname,fmt=&formats);
%patid;
options mprint;

%gencode(gen=&code, outname=&outname);

%macro runcode;
   %if &code^=Y %then %do;
      %runit;
   %end;
%mend;

%runcode;
```

CONCLUSIONS

In this paper we demonstrated a simple method to extend a SAS/IntrNet browsing and reporting application to also be a code-generation application. This method makes use of the internal function of the SAS/IntrNet software and the SAS dictionary. Because of the simple design we propose that this code-generation method can be easily integrated into many SAS/IntrNet applications.

The programs created by this method are saved as stand-alone SAS programs that can be retained for future use in the interactive environment or as independent batch SAS programs. They also provide documentation of programmatic methods employed in generating reports, a feature lacking in some interactive report generator applications, and enables reports to be reproduced at any time, meeting requirements for use in the pharmaceuticals industry.

CONTACT INFORMATION

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

Paul Gilbert
DataCeutics Inc.
1610 Medical Drive,
Suite 208
Pottstown, PA 19464
Phone: 610-970-2333
Email: gilbertp@dataceutics.com

DataCeutics Inc. is a SAS Institute Quality Partner and SAS Institute PharmaHealth Technologies Quality Partner.