

“Specification like this” – a framework for automatic generation of program code and documentation

FDA/PhUSE annual Computational Science Symposium, March 18th - 19th, 2013, Silver Spring Maryland

Poster Number PP06
Marc Andersen,
mia@statgroup.dk

Statgroup ApS, Denmark

Background

- Over the years I have automated reporting of clinical data
- used data definition for ADAM dataset to generate SAS code
 - expanded data display plan to meta data usable for generation of SAS code to generate tables, listings and graphs (TLG)
 - used a simple command language to execute mainly pre-specified reports using different not pre-specified sub groups

Much of the work consist of making datasets and output like the previous with a slight change. With “Specification like this” (SLT) the data sets and outputs are specified using plain text, processed by SAS programs to generate programs and documentation for reporting as exemplified below.

Experience

- it is time consuming to make the initial full specification, changes are fast
- desirable side-effect are high-quality meta-data for data sets and outputs
- documentation and programs are by design synchronized
- consistent naming is easy attained
- nice layout in generated programs is difficult (indentation and line breaks)
- writing programs for code generation is not more difficult than making SAS macros
- code generation can be used to make “executable code” as well as calling standard scripts

Ongoing development

- code generation targeting other languages, such as R
- automated checks of the specifications, such as naming conventions and consistency
- use more direct approach for code generation (StringTemplate)

Conclusion

After using the framework for reporting of two studies and other tasks, I still find it is worthwhile approach supporting “make it like table 14 fast only changing ...”.

Programs available – send me an e-mail.

SLT framework

1. Create specification

```
!SLTDEF TLG
TLGId: E1
Title: Descriptive statistics
OutputId: ${TLGId}
Tableinputdataset: ADAM.ADBL
Tableoutname: Result.Table_${OutputId}
Analysevars: sex/cl age/co weight/co
Colvar: trt01a
Tbyvarvaluelist: Placebo Active
Tbyvarnamelist: ${colvar}
TLGprogramname: Table_${OutputId}
!SLTEND

!SLTDEF TLG
TLGId: E2
!SLTINHERIT E1
Title: @{E1}, subset Age less than 13y
Datasubset: age<13
!SLTEND
```

2. Expand specification into datasets

Dataset for report code generation for descriptive statistics						
Variable name	Report label	Report usage	Report print width	Report just	Report format	Report compute before
Variable_seqno		order	noprint	Right	best8.	
Variable		order	noprint	left	\$32.	
variable_label		order	noprint	left	\$200.	rowabove
statseqno		order	noprint	right	best8.	
statname	Parameter	order	width=24	left	\$64.	
ResColumn1_Placebo	Placebo					
ResColumn1_Active	Active					

3. Create documentation and programs from the datasets

```
/* Code snippet showing code generation */
codeline=cats(catx(" ", "define", variable_name, "/",
    ifc(missing(report_label), " ", quote(strip(report_label))),
    report_usage, report_printwidth, report_just,
    ifc(missing(report_format)," ", cats("format", "=",report_format))
), ";") ;
output;
if report_computebefore="rowabove" then do;
    codeline= cats(catx(" ", "compute before", variable_name,
        "/ style={just=left font_weight=bold}" ),";");
    output;
    codeline=cats(catx(" ", "line @2", variable_name, report_format),";");
    output;
    codeline="endcomp;"; output;
end;
```

4. Generated code

```
/* Code snippet showing the generated code */
proc report data=repdsn missing nofs spacing=1 split='␣' SPANROWS;
    column Variable_seqno Variable variable_label statseqno statname
           ResColumn1_Active ResColumn1_Placebo;
    define Variable_seqno / order noprint right;
    define Variable / order noprint left;
    define variable_label / order noprint left;
    compute before variable_label / style={just=left font_weight=bold};
        line @2 variable_label $200.;
    endcomp;
    define statseqno / order noprint right;
    define statname / "Parameter" order width=24 left;
    define ResColumn1_Active / "Active" display;
    define ResColumn1_Placebo / "Placebo" display;
run;
```

5. Output

Parameter	Active	Placebo
Sex		
F	5 (55.6%)	4 (40.0%)
M	4 (44.4%)	6 (60.0%)
Total (N)	9	10
Age (years)		
N (Nmiss)	9 (0)	10 (0)
Mean (sd)	13.1 (1.5)	13.5 (1.6)
Weight (pounds)		
N (Nmiss)	9 (0)	10 (0)
Mean (sd)	100 (16)	100 (29)

Executable code