

Design Considerations Used to Develop Clinical Reporting Software

Steve Light, DataCeutics

Paul Gilbert, DataCeutics

ABSTRACT

To streamline the production of SAS-generated reports used in regulatory submissions, many companies are purchasing commercial SAS-based clinical reporting software or developing custom SAS-based software. This SAS-based software usually consists of several layers of SAS Macros. This paper discusses the design of the CRT Toolkit[®], commercially available SAS-based clinical reporting software. The discussion will focus on the overall architecture of the software and specific design aspects of the report modules.

INTRODUCTION

Most drug development organizations have two basic needs. The first need is to standardize the reporting and second, to streamline the reporting process. Combined this will increase reporting quality and lead to decrease time-to-market. SAS-generated reports are a key component of most clinical study reports that are included in a regulatory submission. The reporting of most clinical data is similar from study to study and therefore can be standardized. This standardization creates an opportunity for SAS macro-driven reporting. Many companies have developed custom SAS macro-driven reporting utilities; others have chosen to license commercial SAS-based clinical reporting software, such as the CRT Toolkit.

This paper focuses on the overall architecture of the CRT Toolkit, addressing several design aspects of the software. The key design aspects are listed below.

- Report Categorization
- Overall Functional Design
- Overall Technical Design
- Report Module Design

The CRT Toolkit produces listings, summary tables and graphics for use in regulatory submissions. For purposes of this paper, we will limit the discussion to the overall design and the summary table modules only.

REPORT CATEGORIZATION

The CRT Toolkit summary table modules are designed to produce a variety of common summary table formats. They include frequency tables, descriptive statistics tables and combination tables. The table designs are listed below. There are five frequency table designs, seven descriptive statistics designs and two combination tables that display both frequency counts and descriptive statistics. Note that the tables display the same information in different formats.

grp1	grp2	trt1	trt2	trt3
x100	a2000 b3000	n(%) n(%)	n(%) n(%)	n(%) n(%)
y500	a2000 b3000	n(%) n(%)	n(%) n(%)	n(%) n(%)

grp1	grp2	trt1 xxxxyy	trt2 xxxxyy	trt3 x xxxyy
x100	a2000 b3000	n(%)n(%) n(%)n(%)	n(%)n(%) n(%)n(%)	n(%)n(%) n(%)n(%)
y500	a2000 b3000	n(%)n(%) n(%)n(%)	n(%)n(%) n(%)n(%)	n(%)n(%) n(%)n(%)

grp1/ grp2	trt 1	trt2	trt3
x100	n(%)	n(%)	n(%)
a2000 b3000	n(%) n(%)	n(%) n(%)	n(%) n(%)
y500	n(%)	n(%)	n(%)
a2000 b3000	n(%) n(%)	n(%) n(%)	n(%) n(%)

grp1/ grp2	trt1 xxxxyy	trt2 xxxxyy	trt3 xxxxyy
x100			
a2000 b3000	n(%)n(%) n(%)n(%)	n(%)n(%) n(%)n(%)	n(%)n(%) n(%)n(%)
y500			
a2000 b3000	n(%)n(%) n(%)n(%)	n(%)n(%) n(%)n(%)	n(%)n(%) n(%)n(%)

grp1	Treat	Baseline	low	norm	high
x100	trt1	low norm high	n(%) n(%) n(%)	n(%) n(%) n(%)	n(%) n(%) n(%)
	trt2	low norm high	n(%) n(%) n(%)	n(%) n(%) n(%)	n(%) n(%) n(%)

Stat1a					
grp1	grp2	stat	trt1	trt2	trt3
x100	a2000	n	nnn	nnn	nnn
		mean	n.n	n.n	n.n
		std	n.nn	n.nn	n.nn
b3000	n	n	nnn	nnn	nnn
		mean	n.n	n.n	n.n
		std	n.nn	n.nn	n.nn

Stat3a					
grp1	grp2	treat	n	mean	std
x100	a2000	trt1	nnn	n.n	n.nn
		trt2	nnn	n.n	n.nn
		trt3	nnn	n.n	n.nn
b3000	n	trt1	nnn	n.n	n.nn
		trt2	nnn	n.n	n.nn
		trt3	nnn	n.n	n.nn

Stat1b					
grp1	stat	trt1 xxxyy	trt2 xxxyy	trt3 xxxyy	
x100	n	nnnnnn	nnnnn n	nnnnnn	
	mean	n.nn.n	n.nn.n	n.nn.n	
	std	n.nnn.nn	n.nnn.nn	n.nnn.nn	
y500	n	nnnnnn	nnnnnn	nnnnnn	
	mean	n.nn.n	n.nn.n	n.nn.n	
	std	n.nnn.nn	n.nnn.nn	n.nnn.nn	

Stat3b					
grp1	treat	xxx nmeanstd		yyy nmeanstd	
x100	trt1	nnnn.n	n.nn	nnn	n.nn.nn
	trt2	nnnn.n	n.nn	nnnn.nn.nn	
	trt3	nnnn.n	n.nn	nnnn.nn.nn	
y500	trt1	nnnn.n	n.nn	nnnn.nn.nn	
	trt2	nnnn.n	n.nn	nnnn.nn.nn	
	trt3	nnnn.n	n.nn	nnnn.nn.nn	

Stat2a				
grp1/ grp2	trt1	trt2	trt3	
x100				
a2000				
n	nnn	nnn	nnn	
mean	n.n	n.n	n.n	
std	n.nn	n.nn	n.nn	
b3000				
n	nnn	nnn	nnn	
mean	n.n	n.n	n.n	
std	n.nn	n.nn	n.nn	

Stat3c					
grp1	grp2	n	trt1 meanstd	nmeanstd	trt2
x100	a2000	nnnn.n	n.nn	nnnn.nn.nn	
	b3000	nnnn.n	n.nn	nnnn.nn.nn	
y500	a2000	nnnn.n	n.nn	nnnn.nn.nn	
	b3000	nnnn.n	n.nn	nnnn.nn.nn	

Stat2b				
grp1/ grp2	trt1 xxxyy	trt2 xxxyy	trt3 xxxy y	
x100				
a2000				
n	nnnnnn	nnnnnn	nnnnnn	
mean	n.nn.n	n.nn.n	n.nn.n	
std	n.nnn.nn	n.nnn.nn	n.nnn.nn	
b3000				
n	nnnnnn	nnnnnn	nnnnnn	
mean	n.n n.n	n.nn.n	n.nn.n	
std	n.nnn.nn	n.nnn.nn	n.nnn.nn	

Combo1a				
vars		trt1	trt2	trt3
var1	value1	n(%)	n(%)	n(%)
	value2	n(%)	n(%)	n(%)
var2	n	nnn	nnn	nnn
	mean	n.n	n.n	n.n
	std	n.nn	n.nn	n.nn

Combo1b				
vars		trt1 xxxyy	trt2 xxxyy	trt3 xxxyy
var1	value1	n(%)n(%)	n(%)n(%)	n(%)n(%)
	value2	n(%)n(%)	n(%)n(%)	n(%)n(%)
var2	n	nnnnnn	nnnnnn	nnnnnn
	mean	n.nn.n	n.nn.n	n.nn.n
	std	n.nnn.nn	n.nnn.nn	n.nnn.nn

OVERALL FUNCTIONAL DESIGN

The functional design is based upon DataCeutic's experience in performing clinical reporting for clients and our experience developing custom reporting solutions for several pharmaceutical companies. There were several considerations when developing the high-level functional design elements. They address reporting requirements, software design, software functionality and software use. The list below is in order of importance.

The software is designed as a **programmer productivity tool** that is used within a SAS program, which is called in SAS batch mode. It must **improve** programmer productivity by **50%** and support SAS running on the **OpenVMS, Unix** and **Windows** platforms.

The software must be able to produce **80% to 90%** of the reports for most phase 1, 2, 3 and 4 clinical study reports.

The software must be able to meet Common Technical Document (CTD) and FDA regulatory reporting **requirements**.

The software must be **configurable** for a specific client or project. Defaults can be set and stored that define the format of the client's standard report requirements.

Although the reports are standard, they are **flexible standards**, meaning the overall format of the reports are consistent, but flexible within specific portions of the reports. The flexible portions of the reports include:

- Titles and footnotes
- Column/Row variables and labels
- Column/Row widths and wrapping
- Column/Row data ordering
- Treatment labels
- Number/Order of treatment display
- Number of time points or parameters to display
- Paging and groupings

Use **data driven reporting** which allows the content of the data to drive format of the report. Specifically, the software preprocesses the input dataset to create formats, labels, groupings, columns, rows and the ordering requirements of the individual report.

Develop a series of individual **report macros**, specific to the report type and input data structure. **Utility macros** are the building blocks for report macros, containing procedures or code that is used in several report macros. Try to keep the total number of macros to a manageable number, fewer than thirty.

Design the **macro parameters** to be **consistent** across all macros.

Allow the software to be **data model neutral**, meaning there are no variable naming or data structure requirements.

Separate the **pagelayout** (i.e. landscape, portrait, ascii, pdf, titles, footnotes) from the data display and force the same pagelayout for all reports.

Allow for the creation of **one report per output file** and multiple reports per individual SAS program.

Use standard **SAS procedures** such as proc freq, univariate, means and transpose to **process data**, instead of custom, proprietary code, which will ensure the consistency of results.

Calculate and produce **results as a dataset** that can be displayed by the report macro or used in further within a SAS program.

Allow for custom programmed **p-value** results, generated from any SAS-supported statistical procedure, to be included in the reports.

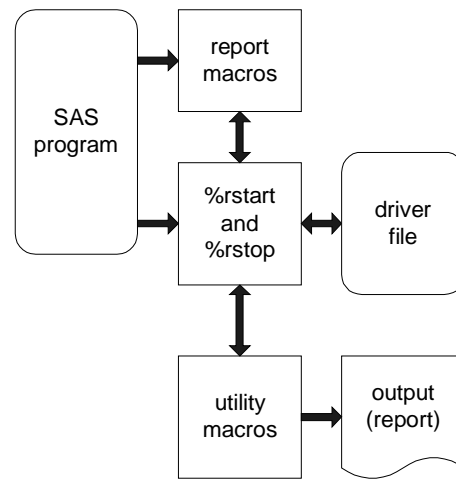
Use **proc report** for formatting and displaying reports. Proc report is a very powerful procedure that provides many features that are hard to duplicate using data _null_. This also allows for utilizing the improvements in SAS technology, such as **ODS**.

OVERALL TECHNICAL DESIGN

The overall technical design makes use of a three-tier design.

The first layer of macros, known as **utility macros**, perform basic

tasks such as interaction with the operating system, file manipulation, managing dynamic formats, manipulation of labels, generating footnotes and titles, performing patient and observation counting, checking dataset status and interaction with the configuration file. These macros are then called and used by the second and third report macro layers.



The second layer of macros is designed to manage the pagelayout and output file creation for all reports. This layer consists of the **%start and %stop** macros. The **%start and %stop** macro pair is used by all report macros (the third layer) and custom reports to format output, control output file naming and destination, control output file type (ascii, rtf, pdf, html), and apply titles and footnotes to reports. They can be used when building programs that do not use the report macros, or to build output files with more than one report procedure output. The **%start and %stop** macro pair creates titles and footnotes, which have been defined in the CRT Toolkit driver file. The driver file can be either a SAS dataset or an ASCII text file.

The third layer of macros, known as the **report macros**, perform the data processing and create the report content - specific to the report type and input data structure. There are report macros for listings, summary tables and graphics. The diagram below illustrates the design. They are discussed in detail in the following section.

<u>Report Macros</u>		
<u>Report Type</u>	<u>Dataset Structure</u>	
	<u>One Record Per Patient</u>	<u>Multiple Records Per Patient</u>
<u>Listing</u>	1 print macro	1 print macro
<u>Frequency Table</u>	1 combo macro	5 freq macros
<u>Descriptive Statistics</u>	1 combo macro	2 stats macros
<u>Graphic Presentation</u>	1 graph macro	3 graph macros

REPORT MODULE DESIGN

This section addresses the CRT Toolkit report macro design. There are several specific report macro design considerations that were an outcome of analyzing the report templates, overall functional specifications and overall technical specifications. We were able to categorize the design points by which specifications related to all reports macros, the frequency report macros and the descriptive statistics macro. Note that most of these points resulted in a user-controllable report macro parameter.

The **first group** of specifications related to **all report macros**, some

of which were incorporated into the %start and %stop macro pair. They are as follows:

- Allow for multiple page layouts and for client-specific configuration of landscape or portrait.
- Display page numbering with syntax of "Page of n" or normal "1,2,3,..". Allow option to start the page numbering with any page number.
- Generate the report titles and footnotes from the macro call or from a central driver file. Define standard titles and footnotes for a project.
- Define the output directory, output filename and output file type from the macro call or from a central driver file.
- Display source program name and output file in the first title or last footnote.
- Ability to pass user-defined pre-report options.
- Display multiple paging variables and allow for using the "by" variable parameters (#byval) in titles and footnotes.
- Selection of a column display variable with default value of the client's standard treatment variable. Automatically calculate appropriate number of columns, determined by data, and fit them to the display area. Wrap columns to a second page in a logical manner.
- Display the value (N=) in column label, where N represents the number of patients in the group.
- Display a second level column variable.
- Calculate the report results and save them as a report dataset that can be displayed or used for merging with other datasets.

The **second group** of specifications relates to the **frequency report macros** only.

- Display up to four levels of row variables, allowing selection of the width of each row variable and automatic wrapping of the data when it does not fit the row width.
- Ability to define total width of stacked row variables.
- Display counts of patients or counts of records (i.e. counts of a type of event) in the column variable. Select counts of most or least value for a patient, for example, most severe event for least related outcome.
- Display counts of events or records occurring in x or % of patients, for example, events that occur in 2 or more patients or medication that occur in 10% or more patients.
- Display an optional "Total" column or "Total" row.
- Define width and format of count "n" and percent (%) in the "n(xx.x%)" format display of the column variable. Allow to zero fill column variable when there are missing values -0 or (0%). Allow client-specified default values for these.
- Allow the data display to be ordered by term (alphabetically), formatted value, ascending frequency or descending frequency. Allow selection of a column to drive frequency ordering.

The **third group** of specifications relates to the **descriptive statistics report macros** only.

- Display up to three levels of row variables, allowing selection of the width of each row variable and automatic wrapping of the data when it does not fit the row width.
- Ability to define total width of stacked row variables.
- Display an optional "Total" column.
- Allow ordering of the data display by term (alphabetically), formatted value, or non-displayed variable.
- Display any statistic produced by proc univariate. Allow customized statistics display order, changing of the statistic name, as well as client or project-specific

defaulting of values.

- Defines skip line variable.
- Allow the display of the number of decimal places in the statistics values to be determined by raw data values.

CONCLUSIONS

The CRT toolkit is a design that addresses the need to standardize the reporting and to streamline the reporting process, thus increasing reporting quality and leading to decrease time-to-market. There are several designs a company can use for developing a comprehensive reporting tool. This paper illustrates one design that was used to develop a comprehensive reporting tool.

The CRT toolkit, based upon this design, has proven to be successful for our own internal use by speeding up our report production as well as external clients as a commercial product where several large and medium-sized pharmaceutical and biotechnology companies license it.

We believe that our three primary goals of designing a **programmer productivity tool**, improving productivity by **50%**, that can produce **80% to 90%** of the reports for clinical study reports, and meeting **CTD and FDA regulatory reporting requirements** have been achieved.

The design, based on component construction, using the power of SAS procedures, has allowed for long-term maintainability. One example is the recent introduction of ODS in SAS version 8.x. Our developers were able to integrate the ODS functionality within the CRT toolkit, by modifying one macro, taking only a few days time.

The layered design, using self-contained, data-driven, report macros for each specific report type, allows us to easily integrate the report macros into applications. Several companies are currently using the CRT toolkit software within SAS/IntrNet applications.

The design, using standard report modules by syntax, used across all report macros, allows for a short learning curve. Most programmers attend a one-day training session and then are up to speed, using the software, in a few days afterwards.

CONTACT INFORMATION

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

Steve Light
DataCeutics
610-970-2333x4723
lights@dataceutics.com
www.dataceutics.com

BIOGRAPHY

Steve Light is a Senior Project Manager at DataCeutics Inc., a leader in clinical information technology. He is responsible for SAS software development and SAS-based clinical programming projects in support of regulatory submissions for the pharmaceutical industry. He has over 17 years of experience with clinical information systems, including SAS-based systems development and validation.

Paul Gilbert is President of DataCeutics Inc., a leader in clinical information technology. His twenty years' experience includes clinical data management, implementing and maintaining clinical systems, designing and developing SAS-based biostatistics reporting systems, managing SAS-based NDA programming support, supporting electronic regulatory submissions, SAS/PHS clinical implementation and support, SAS/IntrNet-based systems, and clinical systems integration.

