PhUSE 2007

Paper TS09

Reporting by Elements – A New Way of Looking at Clinical Reports

Magnus Mengelbier, Limelogic Project, Neuchâtel, Switzerland
Jan Skowronski, Limelogic Project, Malmö, Sweden

The reporting of clinical data within Life Sciences is a common and continuous process. The advent and pressures exerted by requests for short reporting timelines, real time data access and multiple channels of information may impact classical methods of data and information management in a very negative way. We consider common industry processes to generate, manage, verify and document summaries, and how this environment can be optimised. The current form of the programs, output formats and quality assurance requirements require several manual tasks that are time consuming, intensive and may require a larger accumulation of risk of reporting inconsistencies within summaries. We build on new paradigms in generating summaries to keep abreast of requests for real-time access to information while retaining the control over standards, quality and consistency. We also consider how the proposed strategy can be conveniently applicable to efforts in Business Intelligence, information source integration and the next generation of document authoring tools.

INTRODUCTION

The continuous reporting of clinical data is a consistent reminder of the necessity to periodically review and optimise the process and tools. The fundamentals are virtually unchanged governed by industry conventions, standards and regulations. The advent of clinical reporting systems and industry-contributed standards, such as CDISC, may present the vehicle for review, consideration and change.

The current process is well versed within the Life Sciences, with the predominant machinery static in terms of development. We consider the overall fundamental reporting process, quality assurance strategies and standards in order to provide possible optimisation.

The SAS Document Object available in SAS version 9 provides a mechanism for further consideration. We discuss the Document object and how it can refine the current process of generating tabular summaries, listings and figures. We also consider the use of data tables with and as a possible extension to the SAS Document object.

The approach to gain positive benefits through storing statistics and summaries as permanent data sets or data tables are developed in both the context of generating a central report document and disseminating to multiple sources for publication. We also consider the potential benefits if, and when, integration is evaluated.

As we are able to optimise generating the summary, statistics and the final rendering of the presentation and output, the total number of programs can be drastically decreased, the tools standardised, and the process easily integrated.

FUNDAMENTALS OF REPORTING

Clinical reporting is performed in several steps. Source data is extracted from one or more study databases, consolidation and standard derivations are constructed and the final table, listing or figure is produced.

Each step may require several individual programs with possible execution dependencies. It is most common that the extract or import to source data sets is performed by a single, or at most two, programs. If the source databases are of non-standard structures, it may be that the initial extract requires re-mapping or re-coding to a consistent expected source data set structure.

The source data sets are used to further consolidate or derive information such that the final statistic is one or possibly two procedures away. The consolidated, added-value or derived information is most often retained as permanent data sets. Tables, listings and figures are generated based on both the source and derived data dependent on the intent and desired summary.

Each table, listing and figure program would under this process be required to generate both the final statistics and render the output. Consequently, if a table and figure both presents the median for a variable or parameter, it is generated in both

programs. As the program also generates the output, every form of output will have to be generated by the same program or each output specific program would have to generate in parallel the summary or statistic. This web of programs, which can exceed 300 programs for large studies, often requires a great detail of administration, cross comparison and control whilst restricting and reducing efficiency.

The formal Quality Control (QC) procedures or related process checks of above-mentioned outputs, including permanent data sets, are required within many organisations. The data sets can be programmatically compared, creating an efficient and robust QC method. Tables, listings and figures may require a time-consuming manual QC task as well. The manual QC task could be narrowed, or avoided altogether, if the rendering of the final output is performed using a validated set of tools or such that a programmatic compare is possible.

SAS DOCUMENT OBJECT

The SAS Document object, introduced with SAS version 9 (previously experimental), provides some additional features that are interesting to explore. The Document object provides a new ODS destination whereto SAS procedures can send output, similar to that of ODS LISTING, RTF, PDF, etc. The use is similar with ODS destinations, but the capabilities are extended. The full functionality of the ODS DOCUMENT destination and Document object is beyond the scope of this paper, thus only a cursory overview is provided.

The Document object enables any user to *re-play* an output to a specific destination at a later time. For example, when PROC MEANS is executed with both the ODS DOCUMENT and HTML output destinations, the HTML output is generated as well as a permanent SAS output file associated with the ODS DOCUMENT destination. For the argument, say there is a wish to have the same HTML output produced as a PDF file at a later time. Instead of again executing the PROC MEANS procedure once more, the output is re-played with the DOCUMENT procedure and the previously created output is rendered as the PDF from the stored ODS DOCUMENT output.

The re-play effect is obtained as ODS separates the summary generated by PROC MEANS into two halves, data and presentation, in the background, which is now further exposed with the Document object. The ODS DOCUMENT destination retains the two parts of the raw procedure information that defines the data, presentation and any other required information components. In turn, the ODS destinations LISTING, RTF, HTML, etc, in a sense, represents the presentation. Previously, we would employ ODS capture, data sets and output procedures such as PROC PRINT and REPORT to accomplish a similar task.

The Document object is a SAS Item store, e.g. a SAS file, which employs an internal hierarchical structure to organise content that is very similar to that of a file system. In fact, referencing the internal items and values uses a syntax resembling the directory path with folder names delimited by a back-slash (\).

The Document object is designed to encapsulate a procedure, such as PROC MEANS, but not explicitly a specific summary and statistic, such as the baseline Median Systolic Blood Pressure. The net effect is an extended approach to ODS capture with a few caveats.

Our approach coerces the Document object to be more general in storing statistics and summaries. This is a logistical exercise in naming and programming conventions rather than a programming problem.

The approach relies on both the item path and label to store parameter and presentation information. The entry label is restricted to 256 characters, which can be a critical constraint if the approach requires encoding for a wide range of presentation possibilities, labels are naturally long, or if a verbose syntax is employed.

The greatest hinder to use within clinical trial report programs is the lack of support for PROC REPORT, which is a very dominant report utility. We are also not able to extract the Document object information into SAS data sets, so any work-around or techniques based on data sets, e.g. DATA _null_, will fail. PROC TABULATE is supported but the procedure is less used for clinical reporting.

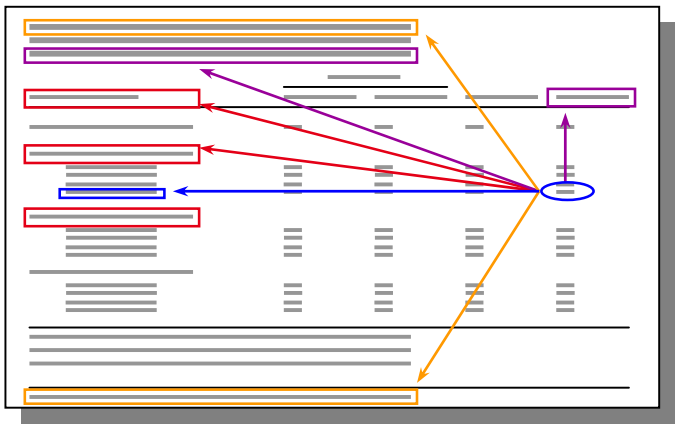
In addition, the Document object file format is currently incompatible with other than the original operating system and no transport format is currently available. This does restrict the distribution, sharing and migration of the Document object to the same platform on which it was created.

We can extend the thoughts of the Document object and possibly hint at a more flexible solution based on data tables that provide a greater flexibility. XML document formats or table storage within a central location or database is a logical alternative, while a SAS data set is another format to consider for smaller study specific implementations.

For the remainder of the paper, we will discuss the use of summary and statistics stored using SAS data sets, but the arguments should be equally applicable to a central database. The XML document format is valid as well, but beyond the scope of this paper.

REVERSING THE TABLE

The process is currently focused on generating the entire table, listing and figure as our primary deliverable. We consider the benefit of a reverse view of our deliverable, concentrating on one or more summaries and statistics that are subsequently displayed as part of a table, listing or figure.



The aim of our proposed process is still remains to deliver a final output in the form of a table, listing or figure. Our discussion centres on the table, but figures and listings with statistics are as applicable.

- The most important is the type of statistic or summary and its corresponding value. The type is usually a keyword, programmatically defined through a standardised list, such as MEAN, MEDIAN, COUNT, PCT, etc. Even the keyword NONSTD(*my statistic*) is a standard of sort.

  The numeric result is stored as both a raw numeric, if applicable, and a formatted character presentation. The character string is a simple means to preserve the presentation format without introducing complex format descriptors. The character string also allows concatenated statistics and summaries, such as *Mean (Standard Deviation)* or *Minimum – Maximum*.

  The label of the statistic is equally important. In the simplest form, the label can represent the preferred label for a standard statistic that can facilitate reporting. The label can also represent a summarised category, say *Female* gender, as the statistic keyword may only refer to a count (COUNT) or percentage (PCT).

- The principal relationship of a summary or statistic is to the parameter, variable and/or category summarised. As we would summarise information located in the source or derived data sets, any reference would only be complete if the data set name is included as well.

  A statistic is seldom presented as a standalone element. For instance, the mean is often accompanied with the number of observations, number of missing, standard deviation, median, minimum and maximum value. The statistics are grouped together to represent a specific parameter, summary group, event sequence, or all of the above. The label has to be flexible to accommodate a very broad range of summary labels and nesting, and yet sufficiently constrained to enforce a consistent convention.

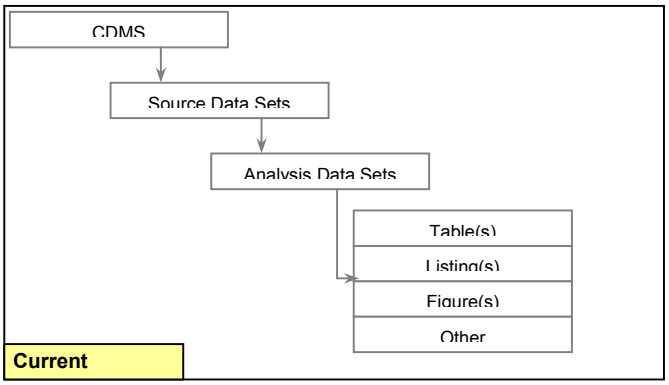
- As our reporting intent is to derive a summary or statistic on values sharing one or more common attributes, group attributes are retained as a reference of context. The principal or primary group attribute is defined to be the table column, for example *Dose Level 10 mg/kg*, in which our statistic is presented. Further selection of specified subsets, such as Safety population or all patients over 60 years of age with at least one treatment administration, is also retained.

- The context may also be retained to document the source and general context of the summary or statistic. Source and derived data sets usually retain a study identifier, but as the reporting event may encompass several studies and specific aims, a greater level of precision may be necessary. For example, a hierarchical directory structure with multiple levels could provide examples of appropriate detail that includes any or all of project, molecule, indication, study, reporting event, etc.

ANATOMY OF THE APPROACH

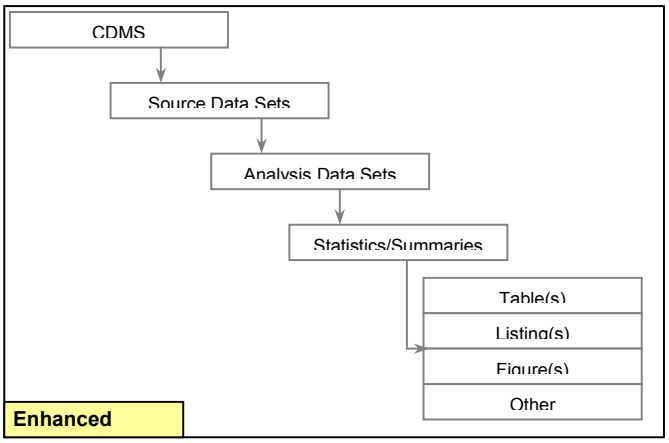
The new approach retains high degree of similarity with the current process, tasks and most common tools. The change, in essence, separates the generation of output in the current process into two distinct and dependent processing steps, statistics and output rendering. As we are required to manage one more deliverable, the new approach can be deemed a step back, an illusion which we will consider further.

As previously discussed, the tables, listings and figures are based on the source and analysis data sets, and their specification. A non-intrusive approach would be to evolve the current process by retaining all elements.



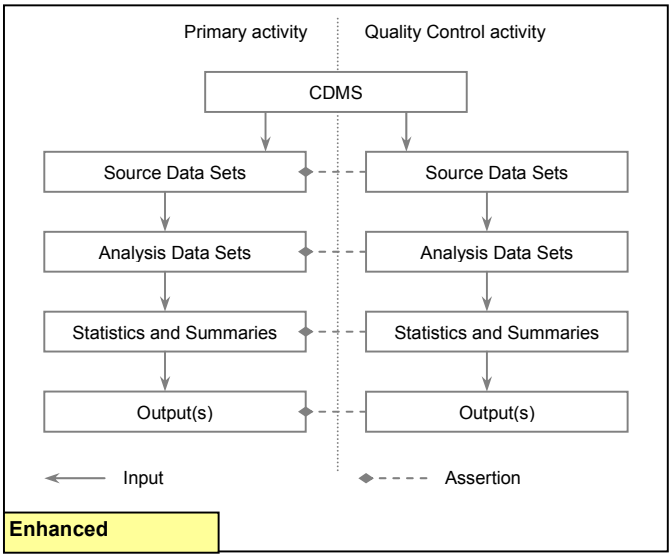
The summaries and/or statistics are most commonly generated as part of the output step within each table, listing and figure program, as applicable. In effect, the code to derive a median may be required in more than one program, if it is presented in different tables and figures.

Consider the output step as two distinct disciplines, whereby summaries and statistics are derived separate from the actual rendering of the table, listing or figure, although still within the same program. The two disciplines of the program is quite similar to the approach of the SAS Document object, consisting of one section for data and one for presentation.



A more efficient approach could be possible if we enhance the current process and separate the derivation of summaries and statistics from rendering the actual table, figure or listing. This would allow us to only generate the summary or statistic once and optimise the program to generate either the summary statistics or rendering, and not be obliged to accommodate both disciplines within the same program code.

For example, consider generating all the statistics for Haematology laboratory parameters in one program and consequently assemble and present the statistics in a table. The approach can be made very simple and efficient as a table and figure presenting the median result over time would only *list* the statistics and not require a parallel derivation.

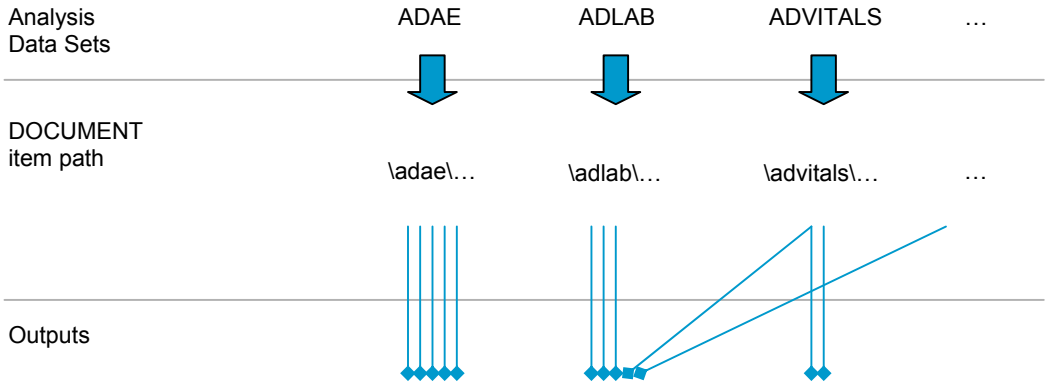


The benefits extend to the Quality Control (QC) process as well, where the largest gains in efficiency are possible. Most organisations will perform assertions for each of the programming steps from study database extracts and source data set to table, listings and figure outputs. The QC process of the extracts, source data sets and analysis data sets are already fairly efficient, as procedures exist to compare data sets (e.g. SAS PROC COMPARE). Efficiently disassembling the outputs programmatically under the current process can be cumbersome or next to impossible staging manual QC as the only surviving option.

The intermediary step of creating statistics and summaries separate from the final rendering of outputs may also enable programmatic QC of the statistics in a very similar manner to derived data sets, regardless if the items are stored as a SAS Document object, data set or in a central database. If we employ validated rendering tools, we may be able to eliminate a large proportion, if not all, of the time-consuming manual QC activities.

USING THE SAS DOCUMENT OBJECT

The SAS Document object, disregarding its caveats, does provide the very basic functionality for permanently storing the raw summaries and statistics generated by statistical procedures. The path reference can be constructed through naming conventions to encode the data domain, population filters, parameters and, of course, our statistic.



The Document object may not be able to include extensive presentation specific information such as pre-formatted values, labels, etc. The label for an item in the DOCUMENT object is restricted to 256 characters, which will limit the degree of nested label information that can be stored. If the total length cannot be constrained to less than the maximum label length, any extensive information would be required to be stored in parallel or coded as part of the rendering.

For example, the mean and standard deviation for Systolic Blood Pressure in the ADVITALS analysis data set could be stored as below list of items.

Element Path	Element Label
\advitals\sbp\safety\trtcd_1\mean	Safety Population@5 mg/kg@Mean@[f=8.1]
\advitals\sbp\safety\trtcd_1\stddev	Safety Population@5 mg/kg@Std Dev@[f=8.2]
\advitals\sbp\safety\total\mean	Safety Population@Total@Mean@[f=8.1]
\advitals\sbp\safety\total\stddev	Safety Population@Total@Std Dev@[f=8.2]

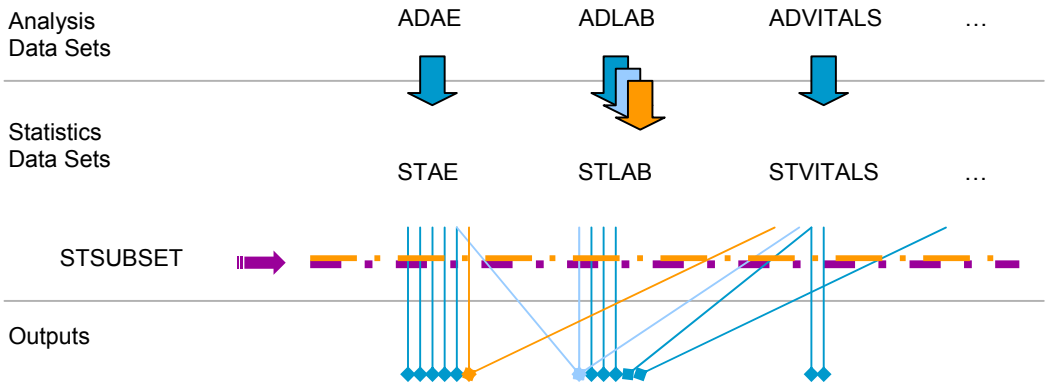
Note the play on naming conventions *safety*, *trtcd_1*, *total* and the corresponding delimited label ('@') that encodes the Safety population subset, first treatment group 5 mg/kg and overall summary group, respectively. A default presentation format (f = w.x) is included in the encoded label as well. Note that path entry names have a maximum length, which can restrict the depth of our naming convention and force extensive use of acronyms, for example *safety60plus* instead of *safetypopulation_overage60*.

In practice, our rendering programs would select the different summaries and statistics, assembling the sequence and generating an output.

USING A STATISTICS DATA SET

The idea of the Document object can be extended and possibly hint at a more flexible solution based on data tables. We briefly discussed the deliverables from a classic reporting process, which usually contains source and analysis data sets as well as the table summary, listing, and figure outputs that presents raw and derived data as well as summary and statistics. We can in addition to the source and derived data sets also provide data sets to contain the summaries and statistics within the suite of deliverables, if applicable.

The effort to include summary and statistics data sets in the list of deliverables is minimal as the data set can be structured to accommodate both the SAS transport file and CDISC ODM formats. Their inclusion can actually greatly simplify generating additional outputs based on already reported statistics and summaries, since only assembly may be required.



As the output step is only required to select, assemble and render the output, we are able to optimise how the summaries and statistics are generated. One approach is to generate summaries and statistics by data set domain, as depicted above. This could imply that all statistics for a data domain, for example Laboratory results, is generated by one single program.

The new data sets are structured to retain all the relevant information about and surrounding our summary or statistic. Following a common analysis data set naming convention, the data set names are prefixed ST… (STatistics). The data set name STSUBSET is reserved for subset definitions, if applicable.

The STSUBSET data set will enable a simple association of a summary or statistic and the underlying observations. In essence, the data set consists of a predefined set of study flag variables associated with the subject ID, assessment identifier and time point variables. The variable label is the subset title to facilitate consistent naming convention and subset references.

The deliverables from a classic reporting process usually contains source and analysis data sets as well as the table summary, listing, and figure outputs. We can in addition also provide data sets to contain the summaries and statistics within the suite of deliverables, enabling a faster QC of the entire package if delivered by a third party or in generating new outputs presenting combinations of previously included summaries and statistics.

As we are able to optimise generating the summary, statistics and the final rendering of the output, the total number of programs can decrease drastically for medium to large studies.

IN CODE

There are several different implementations depending on your preferences, legacy and other factors. A common approach may very well be using a sequence of macro calls performed across different programs. The paper considers the silo approach, whereby a statistics generating program and the resulting ST data set is associated with one analysis data set.

Our aim, in the presented example, is to generate a standard summary table for Height and Weight by Visit for the Safety Population presented for all the available subgroups.

THE STATISTIC

Consider the fictitious macro %stat(), which will be used to generate general statistics. The input parameters include a root path to define context, input data set, a subset definition comprised of the two elements subset flag and corresponding data set.

```
%stat( path = /compound/indication/ia/iss,
       data = derived.advitals,
       subset = ( subset = safety,
                  cntlin = stats.stsubset ),
       group = trtcd _total_,
       by = visit,
       variables = vshght vswght,
       stat = n nmiss mean_sd median min_max q1_q3,
       format = ,
       out = stats.stvitals,
       standards = ( labels = library.stlabels,
                     formats = library.formats )
     );
```

The statistics requested for Height (vshght) and Weight (vswght) consists of two types. The simple case is just a statistic, e.g. Median. The second pertains to composite statistics, such as the mean and standard deviation concatenated and appropriately formatted (mean_sd). By convention, the individual statistics that combine for composite presentations would also be included in the ST data set (output).

We request the statistics by Visit (visit) for the groups defined by the variable Treatment Code (trtcd) and Overall (_total_). No special formats are specified, so any format definitions are retrieved from the standards data set FORMATS. The resulting statistics are appended to the STVITALS output data set.

If we would be requested to generate the above statistics for an additional subset, this can be accomplished by adding an additional subset reference. Note that the entire subset reference is required to match while generating the final output, which would imply the multiple calls to the %stat() macro in order to generate the same statistics for multiple subsets.

```
%stat( path = /compound/indication/ia/iss,
       data = derived.advitals,
       subset = ( subset = safety over60,
                  cntlin = stats.stsubset ),
       ...
       out = stats.stvitals,
       standards = ( labels = library.stlabels,
                     formats = library.formats )
     );
```

THE TABLE

The fictitious %table() macro is the mechanism of preference to generate our table of standard layout. Our titles, footnotes and other specific parameters are retrieved from some central location using the reference name (*reference* parameter).

```
%table( reference = T_HGTH_WGHT,
        path = /compound/indication/ia/iss,
        stats = stats.stvitals,
        subset = safety,
        group = _all_,
        by = ( event = visit, parameters = vshght vswght ),
        sequence = parameter event,
        stat = n nmiss mean_sd median min_max,
        format = _standards_,
        out = M:/compound/indication/ia/iss/blind/output/tables,
        standards = ( labels = library.stlabels,
                      formats = library.formats )
     );
```

Consistent with the %stat() macro, the root path will identify context for the input statistics data set STVITALS.

We select the parameters Height (vshght) and Weight (vswght) to be presented for each Visit (visit) in that order (sequence). Identifying Visit as an event will also sort the table correctly as our standard would first group by parameter and then event.

We select a specific set and presentation order of statistics (stat) excluding the previously generated first and third quartile (q1_q3).

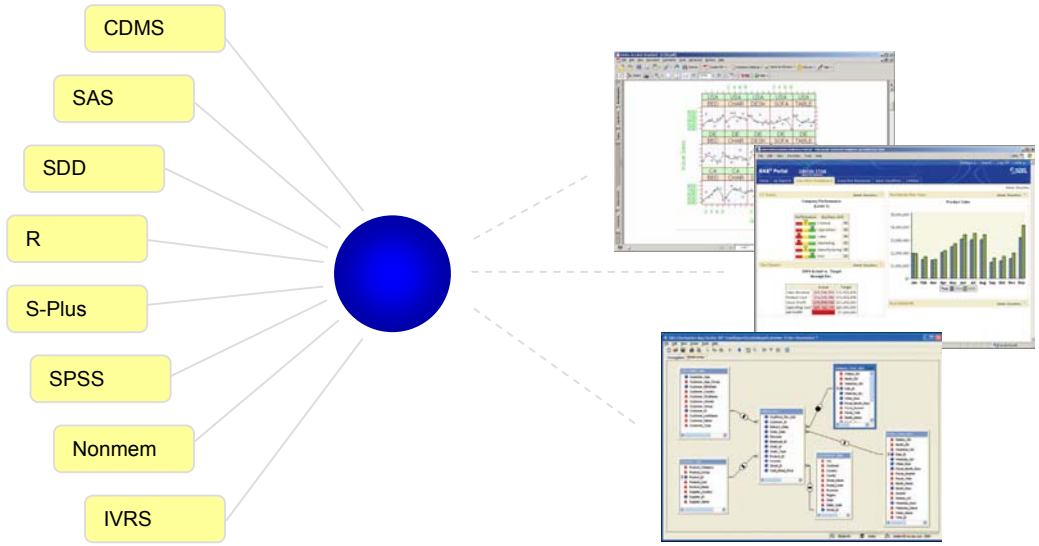
A FIGURE

The parameters are well defined to generate all the appropriate outputs. We can easily use a control data set instead to generate a set of outputs, such as figures for Systolic Blood Pressure, Diastolic Blood Pressure, Pulse, etc. from one macro call. This approach will enable a single location for defining the parameters and standard program templates to easily generate report ready output packages with high degree of efficiency.

```
%figure( reference = F_SBP F_DBP F_PULSE …,
         cntlin = def.figures_sap
       );
```

INTEGRATION

The integrated process is a topic of continuous discussion when new or enhanced systems are considered. The approach presented can easily be extended to encompass other applications than classic SAS and other systems, such as Clinical Data Management System (CDMS), SAS Drug Development (SDD), R, S-Plus, SPSS, WinBugs, Nonmem, etc.



The simple and efficient integration can be achieved because we have already separated the derivation of summaries and statistics from the final rendering of the table, listing and figure. The final element will be to select a central storage location accessible to all the applications and systems.

It is conceivable that any, or all of the above, can contribute summaries and statistics to a single output generated by one application, a custom project portal, company intranet, etc. in a compliant and validated manner. Any restrictions in access to

study results or even a single statistic is very feasible, providing a central secure managed method of publishing and distributing sensitive results

CONCLUSION

The current process of reporting clinical data can be enhanced without significantly altering the approach. The proposed enhancements do retain the fundamentals virtually unchanged as the machinery is to a degree governed by industry conventions, standards and regulations. The gains are obtained from an improved efficiency in reporting fundamentals that sponsor standardisation and improved quality assurance performance.

The SAS Document object is a potential candidate for implementing a simple approach to permanently store statistics and summaries, although the objects deficiencies will disqualify it from more medium to advanced approaches. Further flexibility can be obtained by the permanent STatistics data sets to store all the information and detail surrounding a summary or statistic.

The increased demand for integration and fast secure access is also accommodated as the statistics and summaries can be provided by multiple independent systems best suited for a process while the information is centrally accessible, ready for a report document or dissemination to multiple sources for publication.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. The authors can be contacted through www.limelogic.com/papers or papers@limelogic.com.