PharmaSUG 2013 - Paper AD20

Clinical Reporting by Elements

Magnus Mengelbier, Limelogic AB, Malmö, Sweden

ABSTRACT

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 extend 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 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, 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 a set sequence of 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. Whether the data sets conform to a company standard or even strict or an adaptation of the CDISC Study Data Tabulation Model (SDTM) standards, they are the source for the clinical reporting process.

The source data sets are used to further consolidate or derive information and added as permanent data sets. There are different competing standards, such as mature internal company standards or increasingly CDISC Analysis Data Model (ADaM), such that the final statistic is one or possibly two procedures away. 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 summary or 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 easily exceeds 300 programs for large studies, often requires a great detail of administration, cross comparison and control whilst restricting and limiting 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 while tables, listings and figures most often are exposed to a time-consuming manual QC task. 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.

REVERSING THE TABLE

The focus of the current process is to generate the entire table, listing and figure as our primary deliverable. It is interesting to consider the benefits 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. Our discussion centres on the summary table (Figure 1), but figures and listings are as applicable.

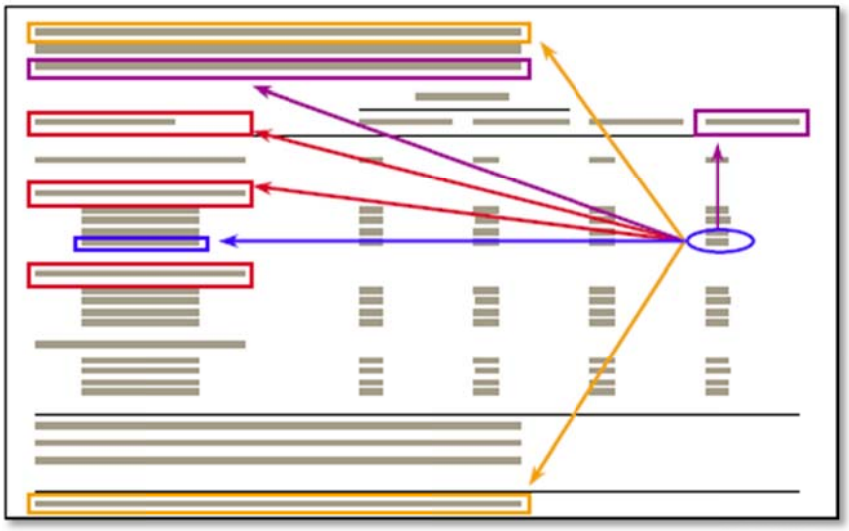

Figure 1. Elements of a summary table

- The most important is the type of summary or statistic and its corresponding value. The type is commonly referred to by 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 has 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 use of a character string also allows concatenated statistics and summaries, such as *Mean (Standard Deviation)* or *Minimum – Maximum*.

  The label of the summary or 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 may also represent a summarised category, say Female gender, as the statistic keyword may only refer to a count (COUNT) or percentage (PCT).

  Further, 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 and the minimum and maximum value. The statistics are grouped together to represent a specific parameter, summary block, 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.

- 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 explicit or implicit reference would only be complete if the data domain is retained in some form.
- 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 a treatment group*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*, is also to be 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.

The reverse view, and the aim of our subsequent process, follows a consistent pattern already used in programming individual outputs and as such will still continue to produce the expected deliverable as all the components of the table, listing or figure are available and assembled. As we discover, a slight change in approach has significant benefits.

ANATOMY OF THE APPROACH

The new approach retains a high degree of similarity with the current process (Figure 2), 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, summary and statistics and subsequent output rendering (Figure 3). As we are required to manage one more deliverables, the new approach can be deemed a step back, an illusion which we will consider further.

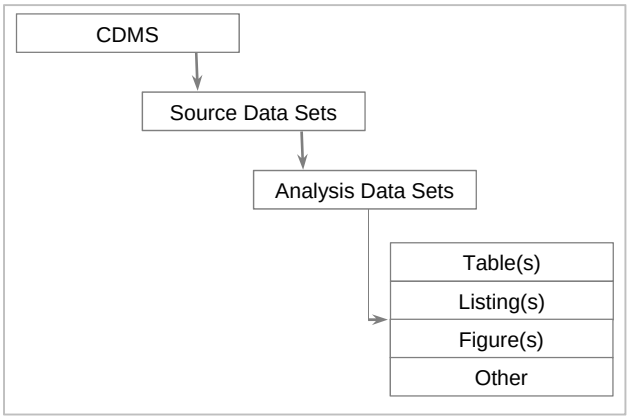

Figure 2. Current standard process

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.

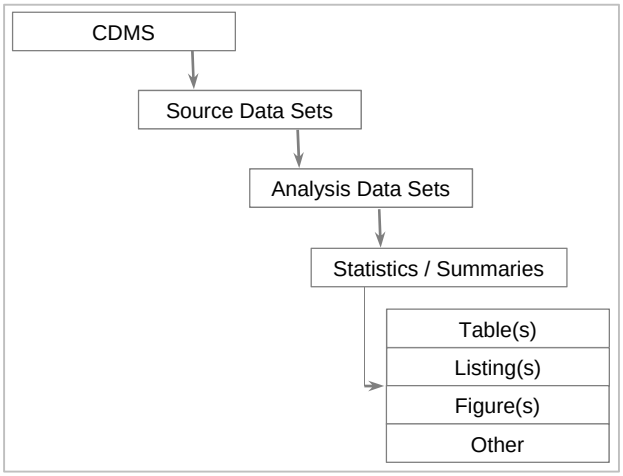

Figure 3. Enhanced process

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 and new opportunities with ODS and the DATA step, a distinct separation of data and 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 and 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 statistical results in a table. The approach can be made very simple and efficient as the underlying program for presenting the median result over time would only render the output and not require a parallel derivation.

The benefits really extend to the Quality Control (QC) conventions and process (Figure 4), 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, such as the SAS COMPARE procedure.

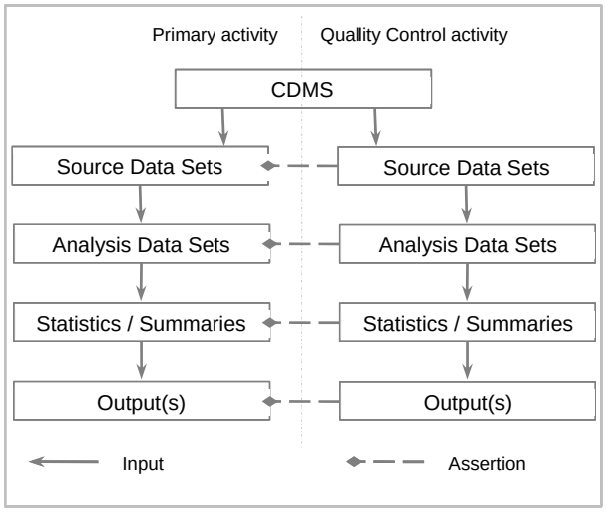

Figure 4. Quality Control convention of the enhanced process

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 deriving summaries and statistics 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 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 A STATISTICS DATA SET

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. In the enhanced process, 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.

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.

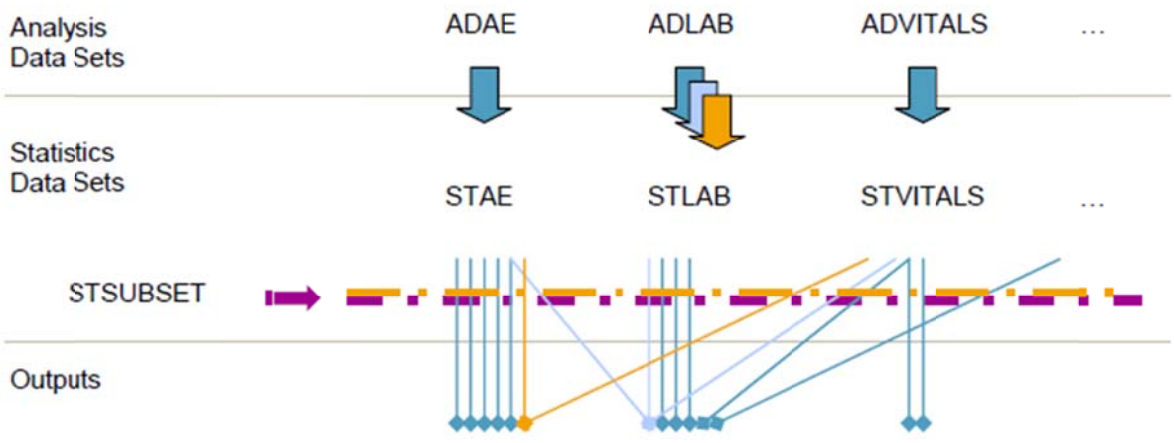

Figure 5. Statistics data sets in process

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 (Figure 5). 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 STxx (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

Consider the example 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 )
      );
```

AND THE OUTPUT

The experimental %table() macro is a good example of the preferred mechanism to generate an output 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).

The parameters can be well-defined to generate all the appropriate output. We can easily use a control data set instead of specifying verbose macro calls in order to generate an output. The resulting effort to generate one or many outputs, such as a set of figures for Systolic Blood Pressure, Diastolic Blood Pressure, Pulse, etc., can be as simple as the code example below.

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

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.

UPDATED STATISTICS DATA SET

Noteworthy, is that the above example does not rely on CDISC standards and there is no logical requirement that it is required to do so. However, the approach has been updated as the use of CDISC ADaM standards have evolved. The first significant change is that the STSUBSET data set is no longer used due to a different approach in how data subsets are defined and referenced. The statistics domain data sets STxx are retained as previously discussed with the exception that population and data subsets are now entirely defined within the STxx data sets.

The guiding principle is that the data for any population or subset has to be defined through the appropriate type of flag. The SUBSET criterion in the STxx data sets support the different level of flags as described in the ADaM Implementation Guide v1.0; subject-, parameter- and record-level flags (Table 1). By convention, the subject-

level flag is defined in ADSL and the parameter- and record-level flags are defined in the applicable domain, which simply implies that deriving statistics on a subset of data remains an instinctive exercise.

The SUBSET variable in the STxx domains uses a simple delimited list of references to the subject-, parameter- or record-level flags. For example, the reference SUBSET = "SAFL:ADPC.PCRFL" is a good example of the short form notation for selecting observations for deriving statistics for the Safety Population and those Pharmacokinetic Concentration (ADPC) records that were selected to be included to generate the statistic estimates.

The notation in the SUBSET variable follows the ADaM standards as introduced by the ADaM Implementation Guide.

Table 1. CDISC ADaM standard convention for flag variables

Level	Notation	Translation
Subject-level	FL (character flag) FN (numeric flag)	FL ="Y" FN = 1
Parameter-level	PFL (character flag) PFN (numeric flag)	PFL = "Y" PFN = 1
Record-level	RFL (character flag) RFN (numeric flag)	RFL = "Y" RFN = 1

The notation was amended to allow for selecting the reverse using the exclamation point ('!') as a prefix, !FL and !FN would imply FL = N and FN = 0, respectively.

INTEGRATIONS

The simple fact that all statistical results may be available in one central location raises interesting opportunities. Producing an integrated summary report of the mean systolic blood pressure at visit 2 across a range of projects or studies is suddenly a very trivial exercise.

At the same, the central repository of statistical results is one foundation for a fully integrated process. Other analysis applications and environments than classic SAS, such as R, S-Plus, SPSS, WinBugs, Nonmem, etc., can easily contribute analysis results for further reporting.

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 with all the appropriate controls, governance and traceability for publishing and distributing sensitive results.

Simple and efficient integration is achieved because we have already separated the derivation of summaries and statistics from the final rendering of the table, listing and figure. Since the statistical results are available, the final component will be to describe how the summaries are to be displayed and rendered.

METADATA

Thus far we have solely referred to the rendering of output using the underlying assumption that it is either a manual programming task or performed by validated tools based on inputs that describe in sufficient detail what statistical results to display and subsequently how (Figure 6).

Figure 6. Detail describing what statistic to display and how

However, there is no obstacle to render output entirely from metadata[2].

The level of detail in the underlying metadata is entirely dependent on the approach exercised. Using templates with few or a finite set of configurable options will involve a small limited set of metadata attributes. Conversely, a completely flexible approach to effectively draw any output within reason will require a more comprehensive and

abstract metadata model. This is fairly accurate regardless if the output is generated as a RTF or PDF document or presented as a part within a portal.

It is most likely that a user will view statistical results as part of an analysis report, or at least as a summary output, rather in its raw data form. To accomplish this, the display and format specifics need to be defined prior to the final output being rendered. An interesting artefact of this sequence is that defining or creating a specification of the desired output will generate the list of the statistics required.

In a naïve view, we have defined the underlying principle of a point-and-click reporting system through the concept of reporting by elements. The next natural evolutionary step is to design and develop a simple interface to edit metadata (Figure 7).

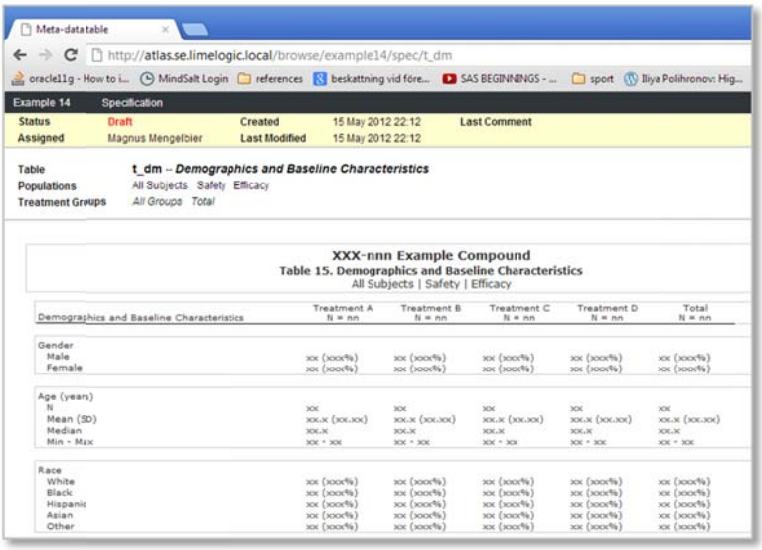

Figure 7. Simple interface to edit metadata

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 use of permanent STatistics data sets to store summaries and statistical results is pivotal to the efficiency in the approach and the opportunities exposed for further integration, whether a single analysis application or an integrated analysis environment. As we have demonstrated, the approach is also not restricted to one single data set standard, applicable to both internal company and the evolving CDISC ADaM standards.

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.

REFERENCES

[1] Mengelbier, Magnus, Skowronski, Jan, 2007, "Reporting by Elements – A New Way of Looking at Clinical Reports", *Proceedings of the PhUSE 2007 Conference*

[2] Mengelbier, Magnus, 2013, "A Simple Interface for Metadata", *Proceedings of the PharmaSUG 2013 Conference*

CONTACT INFORMATION

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

Magnus Mengelbier
Limelogic AB
Jungmansgatan 12
211 19 Malmö
Sweden

E-mail: mmr@limelogic.com
Web: www.limelogic.com