PharmaSUG 2020 - Paper DV-132

Automation of Flowchart using SAS®

Xingshu Zhu, Merck & Co., Inc., Kenilworth, NJ, USA

Bo Zheng, Merck & Co., Inc., Kenilworth, NJ, USA

ABSTRACT

Flowchart diagrams are commonly used in clinical trials because they provide a direct overview of the process, including participant screening, recruited patient enrollment status, demographic information, lab test results, etcetera. Flowcharts are particularly useful when working on Pharmacoepidemiology studies, which require dynamic study designs due to the unpredictability and variation in data sources. In this paper, we introduce a simple SAS macro that allows programmers to create customized flowchart diagrams that fit a customer's individual needs by selecting two different methods and applying them to various types of source data.

INTRODUCTION

Flowchart diagrams are excellent graphical representations of a process or workflow and are commonly used in clinical trials because they provide a direct overview of the process, including but not limited to:

- Participant screening
- Recruited patient enrollment status
- Demographic information
- Lab test results.

The most common flowchart that comes to mind for clinical trials is most likely the CONSORT (Consolidated Standards of Reporting Trials). Flowcharts are also particularly useful when working on Pharmacoepidemiology studies, which require dynamic study designs due to the unpredictability and variation in the data sources. We started the initial process design with the goals of being able to support various types of source data, to significantly reduce the time required to create flowcharts manually in Microsoft Word or PowerPoint, and to automate the counts and contents creation part in SAS. We developed the two distinct workflow process to allow programmers to create customized flowchart diagrams that cover our customers' simple and complex needs.

PROCESS WORKFLOW DESIGN

We started the automation process for creating flowcharts because our flowcharts typically require periodic updates to reflect changes in patient counts or other contents based on the addition of new raw datasets or new cohorts of interest.

The general concept involves creating several elements in SAS programmatically in an organized and customizable way:

- The boxes for each flowchart element.
- The connecting arrows between each element.
- The text, counts, and statistic fields within the boxes.

The end goal was to streamline this previously complicated task into 3 main steps; designing the mockup, creating the template, and outputting the final figure with contents automatically filled in. The first method using a combination of Microsoft Word and SAS programming involves:

1. Creating a Microsoft Word template where boxes, arrows, and fixed text elements are prefilled.
2. Updating the Word template dynamically by unique field names.
3. Using SAS programming techniques to collect the counts and percentages and storing them in

dataset.

4. A simple SAS program reads in both the Word template and the SAS dataset and outputs the finalized flowchart with all the predefined dynamic contents filled in.

The second method using a combination of Microsoft Excel and SAS programming involves:

1. %flowchart0aid.sas macro to create a Microsoft Excel template.
2. %flowchart0test.sas macro to customize, test, and fit the Excel template.
3. %flowchart.sas macro to fill in the flowchart contents from a SAS dataset and output the final rtf table.

We've found both methods to be effective depending on the detail of the initial mockup provided or how comfortable an individual programmer feels about using a Word or Excel based solution for designing the base templates. Overall, the Word template method tended to be faster and more efficient than the Excel method, especially when the dynamic content strings were not too complex to fit and standardized to be used across the studies.

TEMPLATE METHODOLOGY

The first step for creating the flowchart is to have the mockup drawn and ready so the number of elements, their general orientation on the page, and any additional page options are immediately obvious. The mockup can be drawn quickly in Microsoft Word using the Design tab.

The following example mockup is based on a CONSORT diagram that was originally created manually:

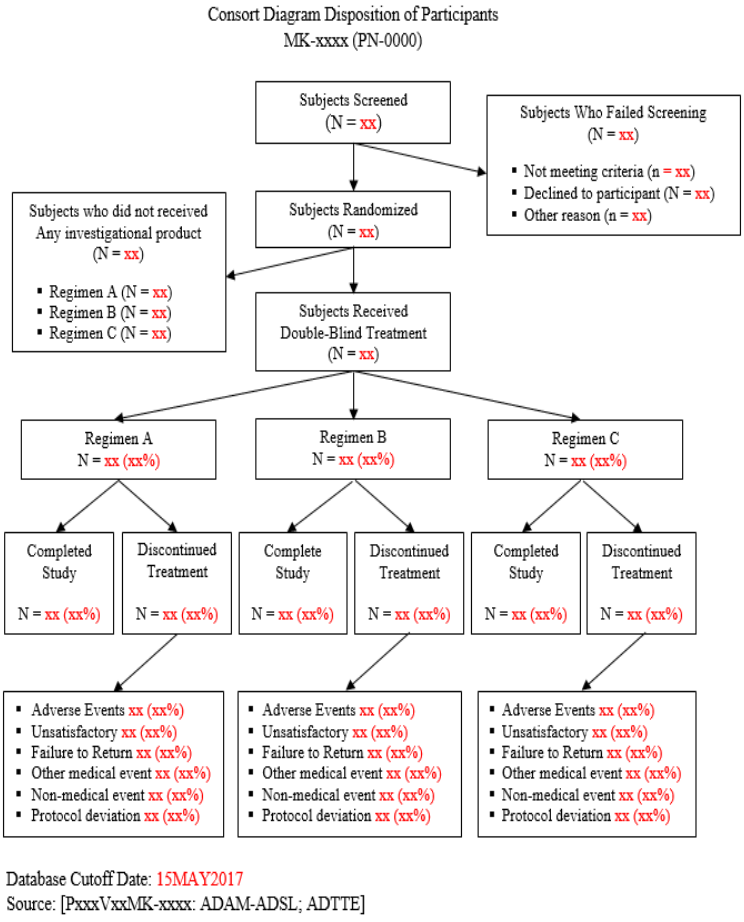


Figure 1. Example Mockup (CONSORT Diagram)

METHOD 1

If your customer has already provided a detailed mockup in Microsoft Word or a similar document like the one above, the easiest method might be to check which elements within the mockup can be dynamically filled in by SAS after reading in the mockup in as the template. This method is best for flowcharts that have relatively set text content and mainly require counts to be filled in, such as the CONSORT diagram example above.

To use this method, replace the red text 'xx' or 'xx(xx%)' with unique field names such as 'U1', 'U2', 'U3', and all the way until you have unique names for every field that need to be filled in. The uniqueness of these field names is important when using SAS to read in the Word template and fill it in the next step automatically. Once the mockup has been updated with the unique field names, create a SAS dataset containing 2 variables, COUNT_NAME and COUNT, through any preferred means or existing macro. Finally read in the both dataset and Word template in SAS, use SAS code to replace the unique field names within the Word template with the contents and counts stored in the SAS dataset, and then read out the completed diagram in a preferred format.

Here's an example for converting the previous CONSORT diagram mockup to a Word template:

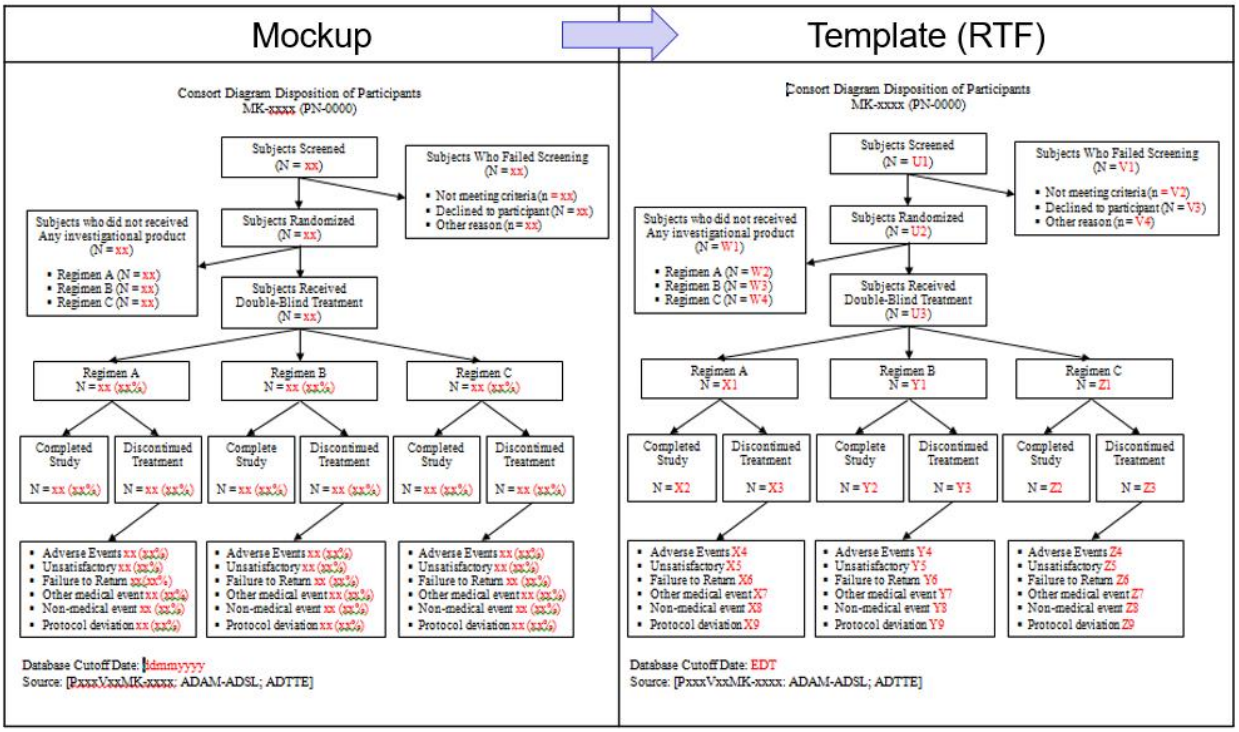


Figure 2. Mockup to Word Template Example

The process for method 1 begins with preparing the mockup in Microsoft Word, replacing the counts or content that need to be updated with unique names that will link to SAS macro variables of the same name. Save this modified mockup file as the Word template. Next, use SAS to create the counts dataset; this dataset will have 2 columns one called COUNT_NAME, that stores the macro variable names that correspond to unique fields within the Word template, and COUNT which contains the basic descriptive statistics such as counts or percentages.

Example:

```
proc SQL;
create table cntds as;
  select 'U1' as count_name, count(*) as count from ADSL;
  insert into cntds
```

```
  select 'U2' as count_name, count(*) as count from ADSL where  randfl='Y';
  insert into cntds
  select 'U3' as count_name, count(*) as count from ADSL where randfl='N';
  /* create more counts as needed */
  insert into cntds
  select 'Z9' as count_name, count(*) as count from ADSL where zzzfl='Y';
  end;
quit;
```

Create a SAS program to read in the Word template with INFILE and the using the TRANSTRN function in a DO loop to replace the target field names in the Word template with the corresponding contents stored in the SAS dataset.

Here's an example of the code:

```
%do i=1 %to &cnt;
  _infile_=transtrn(_infile_
                   ,compress("&&vname&i")
                   ,trim(left("&&&&&&vname&i"))
                   );
%end;
```

Once the DO loop is complete, you can use the FILE and PUT functions to read out the final figure, which will be identical to the Word template but has the contents or counts stored in the SAS dataset instead of the old unique field names such as 'U1' or 'U2.

METHOD 2

If using the Excel method, the initial template does not have to be as complex as the example above. The main things to keep in mind are the number of boxes required, the approximate size of each box, the relative position of each box, and the connecting arrows between each box. The next part of the automation process starts with running the flowchart0aid.sas macro to create the Microsoft Excel base template.

Syntax:

```
%flowchart0aid(rows        = 6
              ,cols        = 3
              ,titles      = 1 /* number of titles in plot-area */
              ,footnotes   = 1 /* number of footnotes in plot-area */
              ,orientation = portrait
              ,outdir       = &wkdir\output
              ) ;
```

If we were trying to recreate the CONSORT diagram from shown in Figure 2, when creating the template, the first step is to figure out how many elements are in the flowchart in terms of columns and rows. For example, when creating the template for a CONSORT diagram consisting of 17 elements, the easiest way to fit this in terms of columns and rows would be 7 columns by 3 rows to create a template with 21 boxes.

Here is a simple example of the suggested edits to the 7x3 starting template to fit the final needs:

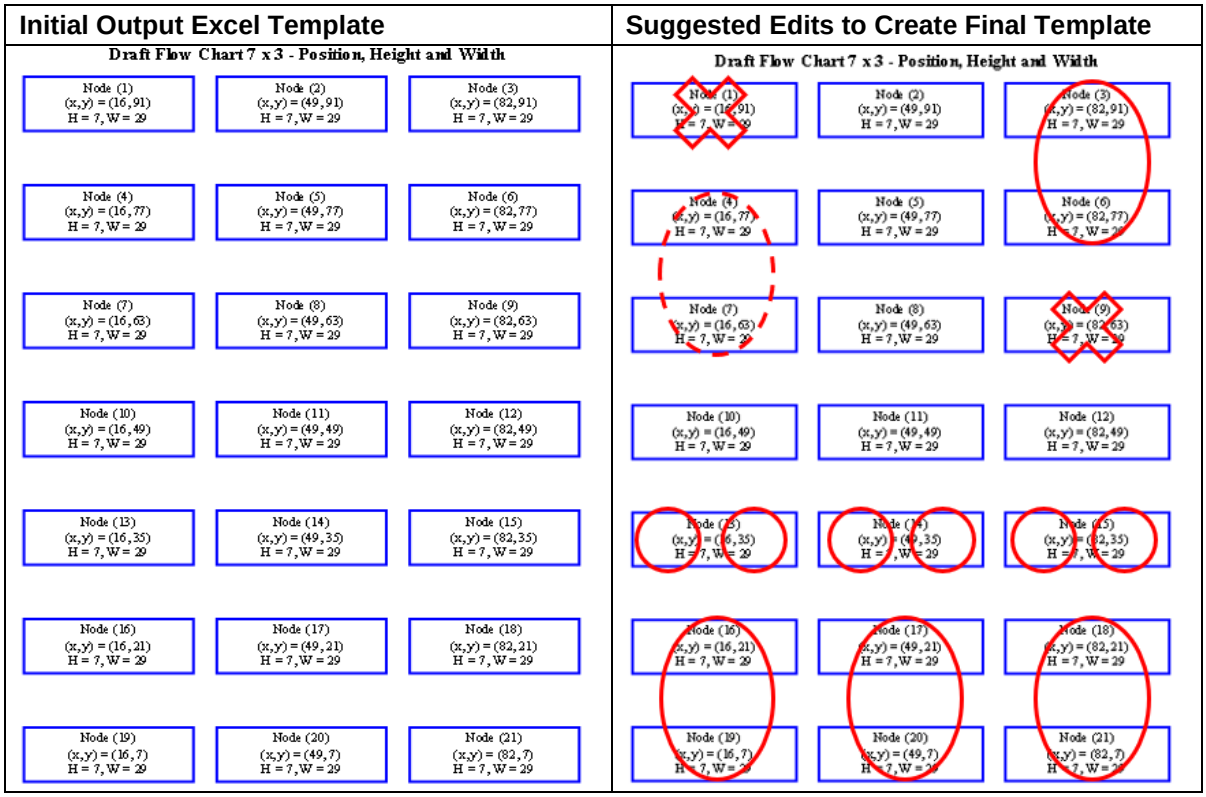


Figure 3. Editing the Initial Excel Template to Final Template

Each Excel template will consist of 7 columns that control separate parts of the overall flowchart:

Col. Name	Description	Type	Required
Node	Node number in the flowchart	Num	Yes
W	Width of the node in the flowchart	Num	Yes
H	Height of the node in the flowchart	Num	Yes
Cx	The X coordinate of the node center	Num	Yes
Cy	The Y coordinate of the node center	Num	Yes
CONTENTS	Leave it blank.	String	Yes
CONNECT	Restrictive enter draws arrow(s) pointing to current node. For example, suppose you are on node 4, then Notation: Draws arrow from bottom center of node 1 to node1: Top center of node 4 node1-R: Right center of node 4 node1-L: Left center of node 4 Flexible enter (x_1,y_1)-(x_2,y_2) draws a line from (x_1,y_1) to (x_2,y_2). (x_1,y_1)>(x_2,y_2) draws an arrow from (x_1,y_1) to (x_2,y_2). Multiple lines/arrows are separated by |.	String	Optional

Table 1. Description of Excel Template Components

Column ‘Node’ determines number of boxes in the flowchart and columns ‘W’ and ‘H’ control the width and height of each box. The boxes can be positioned with columns ‘Cx’ and ‘Cy’ which represent the X

and Y coordinates of the center of each box. The 'Contents' column controls what appears inside each box, this field should be left blank at this stage of development.

Finally, the 'Connect' column controls how connecting arrows are displayed and what elements they connect by using the general set of rules and naming conventions defined in Table 1. For example, to draw an arrow from Box 1 to top center of Box 4, navigate to row numbered 4 in the 'Node' column and enter "node1" into the 'Contents' column. There is also built in flexibility to directly draw connection arrows using (x1,y1) to (x2, y2) using the same (x,y) coordinate system that also controls the relative positioning of each box.

Once the Excel flowchart template is completed, some manual adjusting will be required by the user to get the template to match the layout of the mockup. The second macro flowchart0test.sas can be used to help the user with this process. This macro will read in the Excel template created by flowchart0aid.sas and will output an rtf file showing the current elements and their dimensions.

Syntax:

```
%flowchart0test(in_exl_tmp    = /* input Excel template */
                ,out_file     = /* output rtf test file */
                ,title_out    = /* outside graphic titles, separated by | */
                ,title_in     = /* inside graphic titles, separated by | */
                ,footnote_in  = /* inside footnotes, separated by | */
                ,footnote_out = /* outside footnotes, separated by | */
                ,orientation  = /* portrait or landscape */
                ,height       = /* height of the flowchart */
                ,width        = /* width of the flowchart */
                ,debug        = /* debug (keep temp datasets) */
                );
```

After checking the output appearance with flowchart0test.sas, continue to test fit the template. For the CONSORT example, 2 boxes were removed, 3 split, and 4 combined to match the requirements defined in the mockup. This process can take some time, but once a template is complete, it can be reused or act as a good starting point for other new diagrams.

After some additional fitting for the boxes and arrow elements, the resulting final template is complete:

Initial Output Excel Template

A	B	C	D	E	F	G
node	CONTENTS	CONNECT	w	h	Cx	Cy
1			29	7	16	91
2			29	7	49	91
3			29	7	82	91
4			29	7	16	77
5			29	7	49	77
6			29	7	82	77
7			29	7	16	63
8			29	7	49	63
9			29	7	82	63
10			29	7	16	49
11			29	7	49	49
12			29	7	82	49
13			29	7	16	35
14			29	7	49	35
15			29	7	82	35
16			29	7	16	21
17			29	7	49	21
18			29	7	82	21
19			29	7	16	7
20			29	7	49	7
21			29	7	82	7

Suggested Edits to Template

A	B	C	D	E	F	G
node	CONTENTS	CONNECT	w	h	Cx	Cy
1			29	9	50	88
2		node1	29	9	50	75
3		node1-L	29	15	84	82
4		node2-R	29	15	16	69
5		node2	29	9	50	62
6		node5	29	9	16	49
7		node5	29	9	50	49
8		node5	29	9	84	49
9		node6	13	9	8	35
10		node6	13	9	24	35
11		node7	13	9	42	35
12		node7	13	9	58	35
13		node8	13	9	76	35
14		node8	13	9	92	35
15		node10	29	18	16	16
16		node12	29	18	50	16
17		node14	29	18	84	16

Table 2. Editing the Initial Excel Template into Final Template for the CONSORT Diagram Example

CREATE COUNTS IN SAS

This step can be completed through a SAS macro or program depending on the content requirements needed. For our CONSORT diagram example, some basic counts and display text needed to be pulled from an example ADSL dataset. The created SAS dataset must contain the columns 'NODE' and 'CONTENTS' and that multiple lines are separated by '|'.

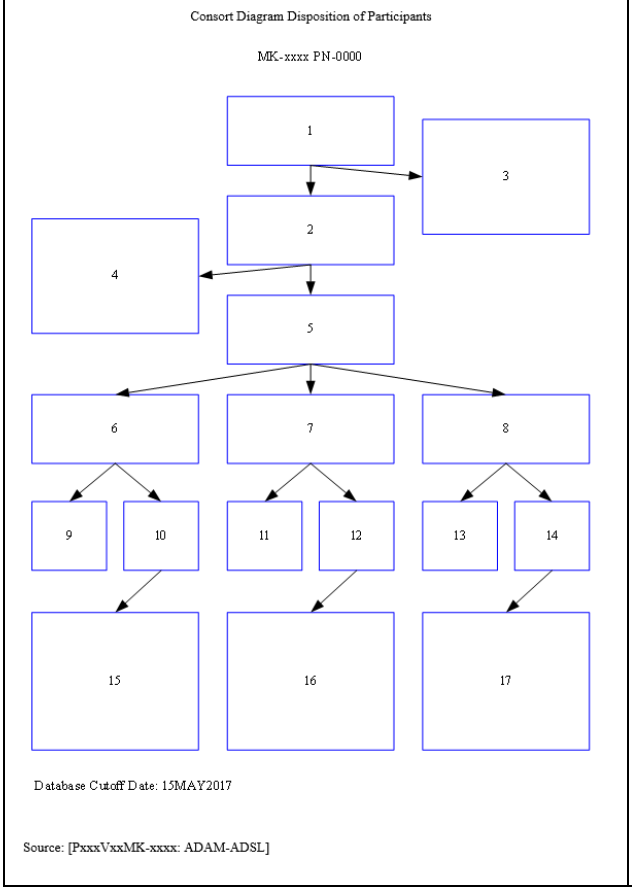
Example of creating counts with proc SQL:

```
proc SQL;
  create table count as select 1 as node,
    'Subjects Screened | (N = '|| put(count(*),3.) ||')' as CONTENTS
    Format=$256. Length=256 from ADSL;
    Insert into count select 2 as node,
    'Subjects Screened | (N = '|| put(count(*),3.) ||')' as CONTENTS
    Format=$256. Length=256 from ADSL where randfl='Y';
    /* any additional statements */
quit;
```

Using PROC SQL to create macro variables was the most intuitive method to accomplish this for the example program, but any existing programs and macros can also be leveraged to make this step more automated. The end goal is to have an output SAS dataset with the necessary details to fill in the 'Contents' column of the Excel template.

On the next step we'll use the completed input template to create the final output diagram shown below:

Final Input Template



Final Output Diagram

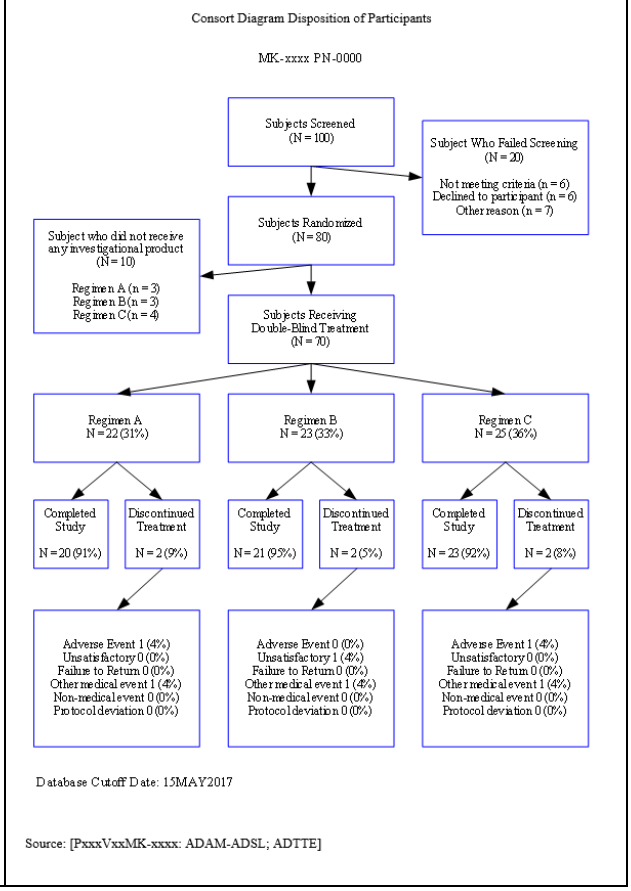


Figure 4. Final Input Template to Final Filled Diagram Output

CREATE THE FINAL FLOWCHART OUTPUT

To build the flowchart, 2 elements are required, the "&in_exl_tmp" Excel file and the "&in_sas_cnt" SAS dataset. These two files are created and modified in the previous defined above. Once these two files are created, it becomes very easy to create the final flowchart by using the macro %flowchart.

Below is the description of the parameters from our flowchart.sas macro:

Parameter	Description	Default	Required
In_exl_tmp	The input excel file which defines the flowchart.		Y
In_sas_cnt	The SAS dataset contains contents/counts in nodes.		Y
Out_file	The output file which save the flowchart.		Y
Title_out	Plot titles out graph area, use | as delimiter for multiple titles.		
Title_in	Plot titles in graph area, use | as delimiter for multiple titles.		
Footnote_out	Plot footnotes out graph area, use | as delimiter for multiple.		
Footnote_in	Plot footnotes in graph area, use | as delimiter for multiple.		
Orientation	Page orientation: portrait or landscape.	Portrait	
Height	Height of the flowchart.	9 in	
Width	Width of the flowchart.	6.5 in	
Debug	Debug (keep temporary datasets)? Y or N.	N	

Table 3. Flowchart Macro Parameters

Flowchart macro call syntax:

```
%flowchart(in_exl_tmp   = /* input Excel template */
          ,out_file     = /* output rtf test file */
          ,title_out    = /* outside titles, separated by | */
          ,title_in     = /* inside titles, separated by | */
          ,footnote_in  = /* inside footnotes, separated by | */
          ,footnote_out = /* outside footnotes, separated by | */
          ,orientation  = /* portrait or landscape */
          ,height       = /* height of the flowchart */
          ,width        = /* width of the flowchart */
          ,debug        = /* debug (keep temp datasets) */
          );
```

At a minimum the first 3 parameters must be filled and point to a valid file or location:

1. The input Excel diagram template (&in_exl_temp).
2. The counts and contents SAS dataset (&in_sas_cnt).
3. A valid out directory to save the .rtf file in. (&out_file).

CONCLUSION

It can be very time consuming and error prone process to frequently update flowcharts by hand and manually enter in new values. We were able to maintain the old process workflow of going from diagram to template to content by automating the process with SAS macros using two distinct methods:

- Method 1 provides a fast and flexible way to easily create the layout and basic descriptive elements using Microsoft Word's point and click interface.
- Method 1 is best used when a diagram or flowchart has relatively simple elements that are updated

such as counts and percentages.

- Method 2 using Excel and the SAS macro set has made it much easier to create and update flowcharts that have a higher degree of variation in design and layout.
- Method 2 does feel and sound more complex at first, but it only takes one or two tries to become accustomed to creating the template and modifying the boxes and arrows through basic X and Y axis positioning and simple notation.

These solutions and the overall process are not only limited to CONSORT diagrams but can be adapted for any variety of custom requirements. The automated process has greatly increased efficiency and has significantly reduced the manual work and overall programming time required across our user groups.

ACKNOWLEDGMENTS

The authors would like to thank Shuping Zhang, Jane Liao, and their other Global Process and Standards colleagues at Merck & Co., Inc., Upper Gwynedd, PA, USA, for their partnership in support of this work.

CONTACT INFORMATION

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

Bo Zheng
Merck & Co. Inc.
351 N Sumneytown Pike
North Wales, PA 19454, USA
bo.zheng1@merck.com

Xingshu Zhu
Merck & Co. Inc.
351 N Sumneytown Pike
North Wales, PA 19454, USA
xingshu_zhu@merck.com