

Paper 1-27

To ODS RTF and Beyond

David Shannon, Amadeus Software Limited, UK

ABSTRACT

The ability to send output to Microsoft Word was the most commonly requested function during the development of the Output Delivery System. Since version 8.1 ODS RTF has been formally available. The Rich Text Format (RTF) specification is a method of encoding formatted text and graphics which is read by Microsoft Word and several other text editors. So now that we have the ability to send output to Word, how can we apply it? What are the advantages over traditional fixed space output? How can we go beyond the documentation to extract some of Microsoft Word's features? This paper will address these questions demonstrating how to define and apply generic appearance to a project and consider the advantages of RTF output to both the programmer and end user. Furthermore it will be shown how to go beyond the documentation by embedding RTF codes to enhance your reports. Finally this paper addresses some practical considerations when generating and distributing RTF files.

INTRODUCTION

For several years as both a statistician and SAS® programmer one of the most consistent problems I have experienced is the distribution of reports and graphics generated from the SAS system. Whilst sending a text file to a recipient is simple enough, for that person to view the text in the intended format requires them to know the page settings, font and font sizes then configure their software before viewing and printing.

Although there are several methods of importing traditional listing output into Microsoft Word via either DDE from within the SAS system or using Microsoft Word itself, this is a cumbersome task requiring additional processing and time.

Sending reports directly from your reporting procedure or reporting data step to a Word document eliminates these problems.

This paper primarily discusses features available in the SAS system version 8.2.

THE MECHANICS OF RTF**WHAT IS "RTF"?**

A Rich Text Format file is a text file using defined control words and symbols that preserve the formatting of text. In the SAS system version 8.2 the RTF specification used creates Word 2000 documents, whilst version 8.1 created Word 97 documents.

TO ODS RTF

Just two additional lines of code are needed to generate a simple RTF file. Consider the following:

```
ods rtf file="report1.rtf";

title "Listing of CLASS Data";
proc print data=sashelp.class;
run;

ods rtf close;
```

The first ODS statement opens the RTF destination; all output generated will be sent to the file report1.rtf. If this file already exists it will be overwritten. It is only when the ODS RTF CLOSE statements are passed that the RTF file is saved to disk.

So end of discussion then?

Perhaps not, if we look at the output created there could be many elements of the default appearance we wish to customise, perhaps enhance specific areas of the report dynamically, or to conform with our companies guidelines, or just to adjust the page settings of the document.

At this point we consider how the output generated is stored inside Word. Taking partial output from the example above, we can see (Figure 1) that a Word table is divided into rows and columns to store the printed output, where each cell contains one element of the report.

Figure 1

Name	Sex	Age	Height	Weight
Alfred	M	14	69	112.5
Alice	F	13	56.5	84
Barbara	F	13	65.5	98
Carol	F	14	62.5	102.5
Henry	M	14	63.5	102.5
James	M	12	57.5	83
Jane	F	12	59.5	84.5
Janet	F	15	62.5	112.5

Titles and footnotes are stored in the header and footer sections of the document, respectively as shown in Figure 2 below:

Figure 2

The default RTF style shades the cells of the header row grey, and makes the header row text bold. All the cell borders are drawn.

When more than one procedure's output is generated in the same file a section break is added to the document. This also allows the titles to change.

WHAT ARE THE ADVANTAGES OF USING RTF?

Compared to the traditional listing output of version 6 the advantages seem quite obvious. However the use of an alternative format for your output may require changes to your standard working practices and therefore you should be able to justify its use.

By using ODS and standard SAS system reporting procedures the programmer no longer needs to write additional code to post process output. In particular distributing graphics in a format

which is easily viewed and printed has previously been cumbersome. In the SAS system version 8.2, SAS/GRAPH® output can be sent directly to a Word document. For example:

```
ods rtf file="graph1.rtf" ;
goptions reset=all;
proc gplot data=sashelp.class;
  plot weight*height=sex;
run;quit;
ods rtf close;
```

Advantages to the reader are most obviously the appearance. Whilst also pleasing to the eye, the ability to change fonts, font sizes, even colouring and shading can be used to enhance various aspects of the report, improve readability and provide an overall professional presentation. As Microsoft Word and other software packages that read Word files are widely used, the end user has a tool to view both text and graphical reports produced from within SAS software.

DEFINING A GENERIC APPEARANCE FOR A PROJECT

Whilst we can use the style options within reporting procedures such as PROC REPORT and PROC TABULATE to define exactly how our Word document will look, it is more efficient and easier to apply generic changes if we define and store a style using PROC TEMPLATE.

By using style statements at individual procedure level we can override the generic styles we defined should we need a specific appearance for individual tables.

Furthermore we can define settings such as the paper size and margins, through the options statement:

```
options papersize=A4 topmargin="1in"
leftmargin="1in" bottommargin="1in"
rightmargin="1in";
```

Supposing we wish to define a look which minimises on drawing border cells in the output. We can use PROC TEMPLATE to inherit a supplied style which closely resembles what we are aiming for and edit its attributes to suit our needs.

```
proc template ;
  define style custom / store=project.styles;
  parent = styles.printer;
  replace fonts /
  'TitleFont2' = ("Verdana, Arial,
                  Helvetica",10pt)
  'TitleFont' = ("Verdana, Arial,
                 Helvetica",11pt,Bold)
  'StrongFont' = ("Verdana,
                  Arial",10pt,Bold)
  'EmphasisFont' = ("Verdana,
                    Arial",10pt,Medium)
  'FixedEmphasisFont' = ("Courier New,
                          Courier",9pt,Italic)
  'FixedStrongFont' = ("Courier New,
                       Courier",9pt,Bold)
  'FixedHeadingFont' = ("Courier New,
                        Courier",9pt,Bold)
  'FixedFont' = ("Courier New, Courier",9pt)
  'headingEmphasisFont' = ("Verdana,
                           Arial",11pt,Bold Italic)
  'headingFont' = ("Verdana,
                   Times",11pt,Bold)
  'docFont' = ("Verdana, Times",10pt);
  style Table from output /
  background = _undef_
  frame = hside
  rules = groups
```

```
cellpadding = 5pt
cellspacing = 0pt
borderwidth = 1pt;
replace color_list /
  "bg" = white
  "fg" = black
  "bgH" = white
  "link" = blue ;
end;
run;
```

There are three key parts to the procedure above. Firstly the name of our style is defined: `define style custom` along with the location to store our style: `/ store=project.styles;`. The "project" part is a libref and "styles" part is the name of a SAS Item Store to contain our style definition. Note that item stores are not visible in the explorer window of SAS.

Secondly we inherit all the styles defined in the printer style: `parent=styles.printer;`

Thirdly there are several statements which alter the attributes of various elements which may appear in our output.

APPLYING A GENERIC APPEARANCE TO A PROJECT

Style definitions are stored in an "item store" which is located in a SAS system library. This can then be made available to our SAS session with the ODS PATH statement.

```
ods path project.styles(read)
      sashelp.tmplmst(read) ;
```

To make the style available to multiple users simultaneously locate the item store in a SAS system library available to all users on the project and give read-only access to the styles. Specifying the default SAS item store after our customised item store tells the SAS system to use the default style attributes if there are any missing definitions in our customised style.

Such an ODS PATH statement is ideally located in a project autoexec file. Note, however, if we need to alter our customised style we must change the access to the item store to update mode:

```
ods path project.styles(update) ;
```

OPERATING SYSTEMS

ODS RTF is available on all operating systems, not just Windows. Note that the RTF is formatted for the environment it is created in. Should you wish to create RTF in one environment and view the RTF in another environment the record separator may need to be altered. For example, if creating RTF files on a mainframe for viewing under Windows then the appropriate record separator syntax for ASCII is:

```
ods rtf record_separator='0D0A'x;
```

Record separator takes a hexadecimal number.

ODS RTF FEATURES

FILE PROPERTIES

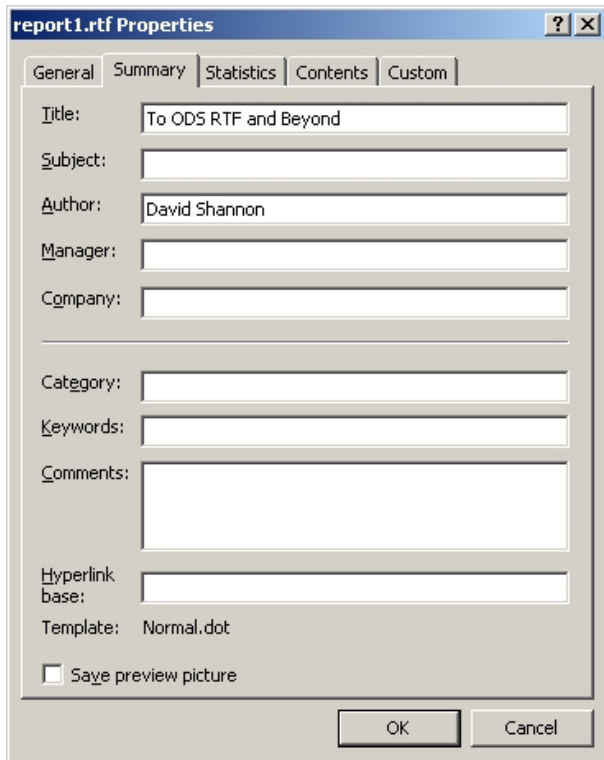
There are several options of the ODS RTF statement which we may use to enhance both the attributes of the document and the contents.

We can set the document author and title properties on the ODS RTF statement as follows:

```
ods rtf author="David Shannon"
        title="To ODS RTF and Beyond"
        file="Report1.rtf";
```

By selecting File -> Properties from within Word, we can see these properties have been set, as in Figure 3 below.

Figure 3



By default the SAS system sets the RTF author as "V8 SAS System Output" and the title as "SAS Version 8". These above options add the title and author information to the document we created.

BOOKMARKS

We can make use of is Word's bookmark facility. When the SAS system creates each new table of output the upper left most cell contains a bookmark, by default, named IDX. Graphic objects generated by the SAS system are also book marked. We can change this for each new table created with the following syntax:

```
ods rtf bookmark="report_1";
```

PAGE BREAKS

The startpage option can be used to control where and when Word places page breaks:

```
ods startpage= now | never | yes | no;
```

Value	Action
Now	Forces a page break now
Never	Prevents all page breaks
Yes	Puts a page break between all outputs
No	Prevents page breaks except around graphics

The startpage option can be used either on its own ODS statement or with other ODS RTF options. For example:

```
ods rtf file="report.rtf" startpage=never;
<procedure>
<procedure>
ods startpage=now;
<procedure>
ods rtf close;
```

We can make use of these options to combine a summary table and graphic on the same physical page:

```
ods rtf file="report1.rtf"
        startpage=no
        style=default;

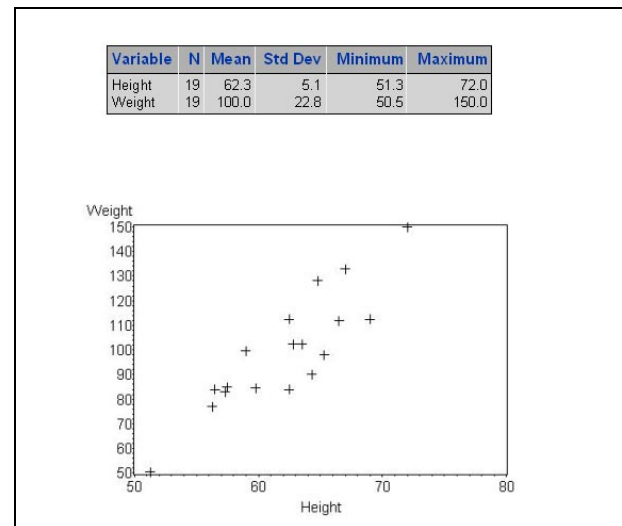
proc means data=sashelp.class nonobs maxdec=1
          n mean std min max;
  var height weight;
run;

proc gplot data=sashelp.class ;
  plot weight * height;
run;
quit;
ods rtf close;
```

There are options we can use to control where titles and footnotes appear. Normally titles and footnotes appear as part of the graphic. However by setting the options NOGTITLE and NOGFOOTNOTE on the ODS RTF statement our titles and footnotes, respectively, will now appear in the header and footer areas of the document.

From the example code above, the document appears as follows to combine a table and graphic on the same physical page as demonstrated in Figure 4 below:

Figure 4



SIMULTANEOUSLY CREATING MULTIPLE RTF FILES

There are situations when we may wish to send output to multiple RTF files simultaneously. We can achieve this by assigning an ID to the open destination.

```
ods rtf(1) file="report1.rtf";
ods rtf(2) file="report2.rtf";
```

When creating multiple files we can also select or exclude output objects to either or both of these files or change the appearance by using the style option on the ODS statement.

For example, consider the output generated by PROC GLM, we may only be interested in specific parts of the results for conveying information, however we may want to retain all the output for documentation. This can be achieved as follows:

```
ods listing close;
ods noproctitle;

title1 h=11pt "Weight = Sex + Age";

ods rtf(1) file="full glm output.rtf"
  style=minimal;

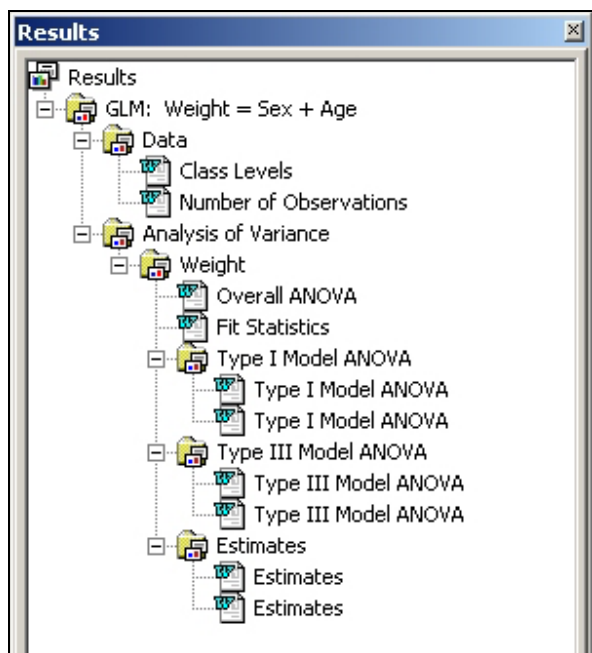
ods rtf(2) file="partial glm output.rtf"
  style=custom;

ods rtf(2) select modelanova estimates;

proc glm data=sashelp.class;
  class sex;
  model weight = sex age;
  estimate 'Male vs. Female' sex 1 -1;
run;
quit;
ods _all_ close;
```

When the results window is expanded (Figure 5) we can see the objects created in two Word documents have two icons:

Figure 5



ESCAPE SEQUENCES

So far all examples of customising text with styles etc. are applied to a complete cell or table. Additionally it is possible to format a section of text within a cell by embedding the appropriate RTF code. Some of these codes are now available via escape sequences which remove the ambiguity of embedding raw RTF codes.

```
data demo;
  length text $400 ;
  text="There are many situations, " !!
  "^S={font_style=italic}from my experience"!!
  "^S={}, particularly in scientific reports"!!
  " where one can take advantage of in-line "!!
  "text features such as those demonstrated "!!
  "here: ^R'\pnhang\par\li720 'To "!!
  "represent powers we can now use x^{super "!!
  " 2}rather than x**2 for just ^S={ "!!
  "font_weight=bold}one ^S={}simple example!";
run;

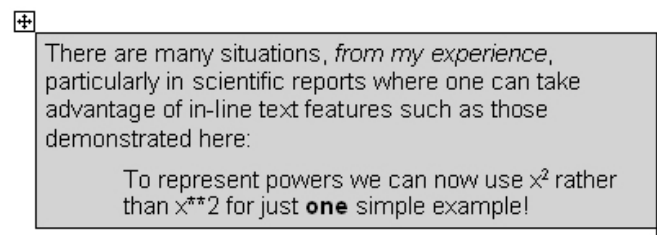
ods escapechar='^';

ods rtf file="escape sequence.rtf"
  style=default;
proc report data=demo noheader nofs
  style={asis=yes frame=box rules=none};
  column text;
  define text / style={cellwidth=10cm};
run;
ods rtf close;
```

This produces the output as shown in Figure 6 below.

So how does this work? Firstly, we define an escape character using the ODS ESCAPECHAR= statement. This tells SAS that whenever it comes to the character we defined, the following text should be treated control words for our ODS destination. In the example above PROC REPORT will see the S={ and R' ' characters defined in our text field. Rather than printing the characters within to the report, the SAS system treats the text as formatting instructions for our output.

Figure 6



Notice the second paragraph begins with a hanging indent. This was achieved by embedding the necessary control codes via the R' ' escape sequence.

The hanging indent is particularly useful for those in the pharmaceutical industry producing reports such as adverse event tables. Here it is usually required that an adverse events' preferred term is presented intended under the adverse events' body system.

AND BEYOND

PAGE NUMBERING

Inserting footnotes as fields such as the "Page X of Y" auto text entry into your document can be achieved by embedding the RTF fields PAGE and Numpages in a footnote:

```
footnote1 j=1 '{Page \field {\*\fldinst PAGE
\*\MERGEFORMAT}} {of \field {\*\fldinst
Numpages \*\MERGEFORMAT} }';
```

The above footnote statement embeds the necessary RTF code into the Word document to display the Page X of Y field. When the document is opened the page numbers may initially appear wrong, particularly in Word 97, this is because Word has not refreshed the fields. When the document is printed the fields will be corrected.

The footer area of a Word document created with the above footnote statement appears as in Figure 7 below. The page numbers are fields in the document:

Figure 7



We can modify the starting page number in our output by using the pgnstartN RTF field, where the suffixed N is the desired start number.

The following sets the start page number at 2:

```
footnote1 j=r '{Page \field {\*\fldinst
{\pgnstart2 PAGE} \*\MERGEFORMAT}}';
```

Note, however, we cannot show the total number of pages within a section until version 9 of the SAS system.

DOCUMENT INFORMATION

It has been requested to me on more than one occasion that the filename and path of the document containing a report output to be printed in a footnote within the report.

As there are several disadvantages of hard coding the path into our output as the storage location may change with distribution of our reports.

With ODS RTF we can embed the field codes into our documents which always show the current path and location of our document. This can be achieved with the following syntax:

```
footnote "{\field{\*\fldinst { FILENAME \*\
Lower\p \*\ MERGEFORMAT }}} ";
```

DECIMAL ALIGNMENT

A commonly requested feature is to align decimal values in reports. This can be achieved by specifying the RTF code "tqdec" and the distance to align the decimal from the left border in a unit called twips (there are 1440 twips in an inch or, one twip is 1/20th of a point).

For example within a PROC REPORT define statement, decimal alignment can be specified with 400 twips from the left cell margin by adding the following style statement:

```
style(column)={protectspecialchars=off
pretext="\tqdec\tx400 "};
```

Producing output such as that shown in Figure 8, below.

Figure 8

Weight (kg)	Female	n	9
		Mean	90.1
		Median	90.0
		SD	19.38

BLINKING TEXT EFFECTS

A further example of enhanced customisation may be to apply a format to given values to make them blink:

```
options orientation=portrait papersize=a4;

title "Highlight Age 16";

proc format;
  value age
    16 = '{\animtext2 16}';
run;

ods rtf file="animated.rtf" style=custom;

proc report data=sashelp.class nofs
  style={protectspecialchars=off};
  format age age.;
run;

ods rtf close;
```

Produces the following output as partially demonstrated in Figure 9: (however when actually viewed in Word the background of the number 16 is actually blinking.)

Figure 9

John	M	12	59	99.5
Joyce	F	11	51.3	50.5
Judy	F	14	64.3	90
Louise	F	12	56.3	77
Mary	F	15	66.5	112
Philip	M	16	72	150
Robert	M	12	64.8	128
Ronald	M	15	67	133
Thomas	M	11	57.5	85
William	M	15	66.5	112

INSERTING PICTURES WITH ODS RTF

A function I frequently performed in earlier releases of the SAS system, was to insert all the graphics generated by SAS into Microsoft Word as individual picture files. Typically these graphics would be generated by several batch jobs and be collated into a single Word document at the end of the job for distribution. Almost inevitably this required some user interaction, or required Microsoft Word to be running additionally.

So can ODS RTF remove that additional step of inserting the picture files into Word and therefore removing any reliance upon your interaction or external software?

Naturally the answer is "Yes". There are two ways in which we can achieve this. The most efficient of which is in version 8.2. For example:

```
data imagelist;
list='^S={preimage="C:\Program Files\SAS Inst
itute\Shared Files\Images\color01medical.jpg"
}';
run;

ods escapechar='^';

ods rtf file="ImageList.rtf";

proc report data=imagelist noheader nofs
style={frame=void rules=none
protectspecialchars=off};
define list /style={cellwidth=10cm} center;
run;

ods rtf close;
```

The graphic appears in a cell of its own in the resulting Word table.

There are three key parts to the example above.

Firstly we created a data set which contains the ODS ESAPECHAR= facility to add in-line formatting. In this case we have used PREIMAGE style option `^S={preimage= }` to insert an external graphic file. Secondly we tell the SAS system what the escape character is, with the ODS ESCAPECHAR= statement. Here the caret is chosen because this character does not, and we would not expect, this to appear in the path name of our graphic file and therefore cause confusion. Thirdly the data set is reported. PROC REPORT is chosen as this allows this width of the column to be controlled. By default the Word table may not be wide enough to accommodate the full graphic, hence we can widen it if required.

In version 8.1 which precedes the ODS ESCAPECHAR facility, we can still achieve the same result by embedding the RTF field to insert a picture. This is the INCLUDEPICTURE field.

The syntax is as follows:

```
var='{ \field { \* \fldinst INCLUDEPICTURE
"path\filename" \* MERGEFORMAT } }';
```

One feature of this method is that a backslash must be translated into four backslashes to be correctly interpreted by Microsoft Word, for example: "C:\mygraph.jpg" becomes "C:\\\\mygraph.jpg".

This can be easily converted into a macro call, which takes the path and filename to your graphic, adds the necessary RTF codes and translates the backslashes for you.

CREATING SCIENTIFIC EQUATIONS

To correctly present formulae or perhaps something as simple as a square root sign has previously been difficult if not impossible using traditional monospace output. As with many of the examples seen here with ODS RTF, we can take advantage of Microsoft Word features. By inserting a field called EQ we can construct equations in our Word document.

The following code creates three records in a data set which are then sent to a RTF file via PROC REPORT. Each record uses a different style of equation.

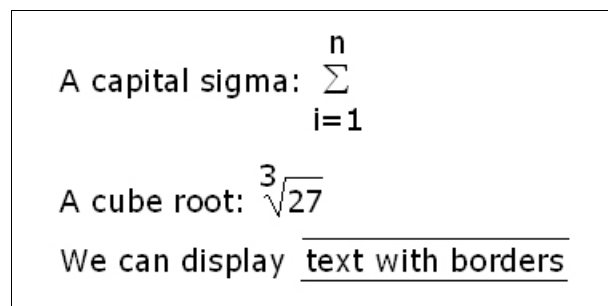
```
data t;
length x $150;
x="{A capital sigma: \field{ \* \fldinst {EQ
\\I \su(i=1,n, ) } } }";
output;
x="{A cube root: \field{ \* \fldinst {EQ
\\R(3,27) } } }";
output;
x="{We can display \field{ \* \fldinst {EQ
\\x \to \bo(text with borders) } } }";
output;
run;

ods rtf file="report1.rtf" ;
proc report data=t noheader nofs
style={protectspecialchars=off
frame=void rules=none};
run;
ods rtf close;
```

When viewed in Microsoft Word we see the output as shown in figure 10.

Note the use of the `protectspecialchars=` style option above. This is necessary when embedding RTF control codes as we need to tell the SAS system to treat the symbols such as braces and slashes as control words and not simply as text.

Figure 10



To view the full list of equation switches and examples; open Word (2000) and press F1. Select the Index tab from the help window and type in "EQ" and press search. The list of equation field codes will appear.

TABLES OF CONTENTS

One further practical feature would be to insert a table of contents at the beginning of a document, which may contain several tables and/or graphics. As a table can have embedded hyperlinks we can navigate our way around the document in a similar way to a web page.

Whilst Word can create a table of contents based on many types of index, it cannot create them based on bookmarks, which are already embedded in our output.

We can however improvise. The following code creates a listing, a summary table and a graphic; where the first data step and PROC REPORT insert fields of links to the bookmarks.

```
data t;
  length toc $100 pageno $100;
  toc='{field {\*\fldinst HYPERLINK \l
"Listing"}}';
  pageno='{field {\*\fldinst PAGEREF Listing
\\h}}';
  output;
  toc='{field {\*\fldinst HYPERLINK \l
"Summary_Table"}}';
  pageno='{field {\*\fldinst PAGEREF
Summary_Table \\h}}';
  output;
  toc='{field {\*\fldinst HYPERLINK \l
"Graphic"}}';
  pageno='{field {\*\fldinst PAGEREF Graphic
\\h}}';
  output;
run;

ods rtf file="bookmark.rtf"
  style=custom
  author="David Shannon"
  title="To ODS RTF and Beyond";

proc report data=t nowd
  style={protectspecialchars=off
  rules=groups
  frame=hsides};

  column toc pageno;
  define toc / display 'Table' left
  style={cellwidth=5cm
  just=left};
  define pageno / display 'Page'
  style={cellwidth=2cm
  just=right};
run;

ods rtf bookmark="Listing";

proc report data=sashelp.class nowd;
run;

ods rtf bookmark="Summary Table";
proc tabulate data=sashelp.class ;
  class sex;
  var height weight;
  table (height weight)*mean, sex;
run;

ods rtf bookmark='Graphic';
goptions papersize=A4 rotate=landscape
  device=activex;
```

```
proc gplot data=sashelp.class;
  plot height * weight = sex;
run;
quit;

ods rtf close;
```

The table of contents is produced on the first page as shown in Figure 11, below:

Figure 11

Table	Page
Listing	2
Summary Table	4
Graphic	5

Word must refresh the fields before the page numbers display the correct page numbers in the document. This can be achieved by selecting the output and pressing F9 from within Word, or by printing the document.

Clicking on the blue hyperlink or page number jumps to that output. Word then displays a web toolbar which allows navigation around the document, in a similar way to a web page.

PRACTICAL CONSIDERATIONS

When creating reports which span several pages, the size of the resulting document should be considered. RTF files become large relative to traditional listing output, hence consideration should be given to storage space and the media used for distributing your files.

Features within Microsoft Word vary between versions, if generating output to ultimately be viewed on older version of Word or other editor, consideration should be given to which features will and will not be honoured.

CONCLUSIONS

Compared to the traditional listing output we can quickly create reports which are both professional in appearance and easy to distribute.

There are benefits to both the programmer in terms of reduced manipulation of output and also to the user for enhanced and professional appearance.

Although initially complex it is possible to extract some of Microsoft Word's features to further enhance RTF output.

The life of ODS RTF is young and there are several areas which could be developed such as improving the ability to use Word's fields, the ability to merge cells and control the drawing of specific cell borders.

REFERENCES

Schellenberger, Brian T. (2000), "*Presentation-Quality Tabular Output via ODS*", SAS Institute Inc.

CONTACT INFORMATION

I very much welcome your feedback on this paper or your thoughts on the topic of ODS RTF. Contact the author:

David Shannon
Amadeus Software Limited
Orchard Farm
Witney Lane
Leafield
Oxfordshire
OX29 9PG
United Kingdom

Phone: +44 (0)1993 878287
Fax: +44 (0)1993 878042
Email: david.shannon@amadeus.co.uk
Web: www.amadeus.co.uk

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

© Amadeus Software Limited, 2001-2002.