SAS-Ergebnisse in Word einbinden und aktualisieren

Christoph Klein
Landesbank Baden-Württemberg
Am Hauptbahnhof 2
70173 Stuttgart
Christoph.Klein@LBBW.de

Steffen Melang
Landesbank Baden-Württemberg
Am Hauptbahnhof 2
70173 Stuttgart
Steffen.Melang@LBBW.de

Zusammenfassung

Schreibt man Berichte in Word, reift das Ergebnis stetig: Oft muss man Tabellen und Grafiken aktualisieren, weil sich die Analysen zwischendurch geändert haben. Dies ist leicht möglich und wird von Word unterstützt, wenn die Tabellen und Grafiken aus RTF-Dateien stammen und mit Textmarken versehen sind. Wir zeigen, wie wir RTF-Dateien und Textmarken mit dem Output Delivery System (ODS) von SAS erstellen und verwenden.

Im Detail erläutern wir:
• Anpassung des ODS-Tagsets RTF
• Berücksichtigung von by-Verarbeitung
• Zuschneiden von Grafiken
• Beschriftung und Nummerierung der Tabellen und Grafiken im RTF

Schlüsselwörter: ODS-Tagset RTF, Word

1 Einführung

Wir verfassen regelmäßig Berichte in Word, in denen wir Ergebnisse aus SAS verwenden, vor allem Tabellen, die mit Reportprozeduren erzeugt wurden, und Grafiken. Wir beginnen den Bericht zu schreiben, bevor alle Ergebnisse endgültig vorliegen. Diese können sich im Laufe der Zeit ändern, dann muss auch der Bericht angepasst werden.

Unsere Anforderungen sind:

- Aktualisierung im Bericht bei Wunsch, aber ohne manuelles Copy und Paste
- flexible Auswahl der Ergebnisse, die in den Bericht einfließen
- automatische Beschriftung in Word mit fortlaufenden Nummern je Abschnitt
- automatische Beschriftung in Word für jede by-Gruppe
- Verwendung von Schriftarten des Corporate Designs in allen Ergebnissen
- editierbare Tabellen
- Vektorgrafiken

Dateien im Word-Format docx kommen zur Lösung nicht in Frage, da sie komplizierte Bündel von XML-Dateien sind; RTF-Dateien sind dagegen einfacher zu erzeugen.

Unsere Lösung: SAS erzeugt eine RTF-Datei mit Textmarken um jedes berichtete Ergebnis und weiteren Textmarken um dessen Beschriftung. Wenn wir ein Ergebnis im Bericht verwenden wollen, binden wir es so in den Bericht ein: Einfügen → Objekt → Text aus Datei; als Bereich muss die Textmarke des Ergebnisses angegeben werden. Dann wählen wir „als Verknüpfung einfügen“, denn nur so können wir das Ergebnis später aktualisieren.

Eingefügte Ergebnisse aktualisieren wir über die Knopf „Felder aktualisieren“. Anpassungen an Tabellen, Grafiken und Bildunterschriften gehen dabei verloren, wenn sie im Bericht und nicht in SAS vorgenommen wurden.

2 Technische Umsetzung

2.1 RTF-Dateien und Textmarken

RTF steht für Rich Text Format. RTF-Dateien sind Textdateien, die Anweisungen enthalten, die das Aussehen des Textes verändern. Diese Vorgehensweise heißt Textauszeichnung (Markup) und ist ähnlich wie bei HTML und LaTeX. Außerdem können RTF-Dateien mit speziellen Befehlen auch Binärdaten wie beispielsweise ein Bild enthalten. Das folgende Beispiel zeigt, wie eine RTF-Datei funktioniert.

Tabelle 1: Markups in einer RTF-Datei

Text mit Markups	Darstellung
{\rtf1 Hallo Welt! \line {\i Text} kommt \b{\i in \i0 vielen \b0Formen}. \par \i0 Beste \b0Grüße. }	Hallo Welt! *Text* kommt ***in*** **vielen** Formen. **Beste** Grüße.

Steuerzeichen für die Textauszeichnung beginnen mit \. Eine Liste findet sich in [1], wir geben nur wenige Beispiele:

- \i und \i0 schalten kursive Formatierung ein und aus.
- \fs11 stellt Schriftgröße 11 ein.
- \line erzeugt einen Zeilenumbruch, \par einen Absatz.
- \field fügt ein dynamisches Feld ein. Wir verwenden es später für die Nummerierung mit aktuellem Kapitel und laufender Nummer je Kapitel.
- Die Zeichen * leiten einige Steuerzeichen ein, die nach 1987 eingeführt wurden.

Eine Textmarke wird im RTF-Dokument durch die folgende Sequenz definiert:

{*\bkmkstart MyBookmark}
Mein Text, Tabelle oder Bild innerhalb der Textmarke (englisch: Bookmark).
{*\bkmkend MyBookmark}

Durch die \bkmkstart-Sequenz wird hier die Textmarke MyBookmark gestartet. Alles zwischen dieser öffnenden Sequenz und der dazugehörigen Endsequenz \bkmkend ist Teil der Textmarke.

Wir verweisen in Word oft auf Tabellen, Grafiken oder andere Abschnitte: „Tabelle 5.1 zeigt, dass …“ oder „ … siehe Kapitel 2 Theoretische Grundlagen“. Diese Referenzen erstellen wir mittels Einfügen → Querverweis. Dabei erzeugt Word eine ausgeblendete Textmarke um die Beschriftung der Tabelle, Grafik oder um die Überschrift des Abschnitts. Der Name der ausgeblendeten Textmarke beginnt mit _Ref. Ausgeblendete Textmarken können aufgelistet werden, wenn man den entsprechenden Haken im Dialog unter Einfügen → Textmarke deaktiviert.

Verweist man jedoch auf ein verknüpftes Objekt, das aus einer anderen Datei stammt, legt Word keine ausgeblendete Textmarke an; eine vorhandene ausgeblendete Textmarke kann Word dennoch verwenden. Um ein verknüpftes Objekt referenzieren zu können, muss die Textmarke für die Referenz bereits vorhanden sein. Daher erzeugen wir die Textmarken für die Referenzarten „Gesamt Beschriftung“ und „Nur Kategorie und Nummer“ bereits mit SAS im RTF, ihre Namen beginnen wir mit _Refa und _Refb (siehe Events Table und New_Image_Start in Abschnitt 2.3).

2.2 RTF-Ausgabe mit ODS

So erzeugt man eine RTF-Datei mit dem Output Delivery System (ODS) von SAS:

```
filename myfile "myfile.rtf";
ods tagsets.myRTF file = myfile;
... Code zur Ausgabe ...
ods tagsets.myRTF close;
```

Das Tagset myRTF definieren wir in Abschnitt 2.3. Um später die einzelnen Tabellen und Grafiken in Word mit Namen ansteuern zu können, ergänzen wir Berichtsprozeduren um die content-Option und Grafikprozeduren um die Description-Option. Ziel ist ein Name nach diesem Schema: <Kürzel des Codeblocks>_<Kürzel des Ausgabeobjekts>_<By-Variablenliste>. Das Kürzel für den Codeblock ist nicht zwingend, doch werden die Namen damit übersichtlicher.

```
proc report    data = datenbasis   contents = "RNBT_RC";
     by segment rating_typ;
     [...]
run;

proc gplot     data = datenbasis;
     by segment rating_typ;
     plot [...]
     / description = "RNBT_GRZEIT_#byvar1=#byval1 _#byvar2=#byval2"
     ;
run;
```

Proc Report fügt die by-Variablen automatisch an den Inhalt von content an, bei den Grafikprozeduren müssen wir sie angeben und mit Leerzeichen trennen. Falls Label vorhanden sind, werden sie in den Namen verwendet. Die Namen nach diesem Schema können länger als 40 Zeichen werden, was für Word jedoch zu lang ist. Daher erstellen wir dynamisch das Format ods_byfmt, das lange Namen sinnvoll kürzt. Außerdem erstellen wir das Format ods_beschriftung, das die Kurznamen auf die gewünschte Beschriftung abbildet. Dies führen wir hier nicht weiter aus.

2.3 Anpassungen des Tagsets RTF

SAS-Prozeduren für Berichte und Grafiken rufen bei der RTF-Ausgabe zwei ODS-Events auf: Table (Proc Report) und New_Image_Start (Proc GPlot, GChart und GBarline). Diese rufen direkt und indirekt andere Events auf wie Branch, Leaf und ByGroup. Wir ändern diese Events und führen das neue Event Textmarke ein (vgl. Tabelle 2).

Tabelle 2: wichtige ODS-Events für die RTF-Ausgabe

Event	Funktion
Textmarke	Erzeugt den Textmarkennamen und die Beschriftung mit Hilfe der Formate ods_byfmt und ods_beschriftung.
AddFont	Fügt dem RTF-Header eine Schriftart hinzu und speichert deren Nummer in einer globalen Variablen.
Anchor	Setzt Textmarken für das Inhaltsverzeichnis.
Branch	Speichert content-Angabe und by-Variablen von Proc Report und Proc Tabulate in globalen Variablen.
ByGroup	Speichert, ob eine by-Gruppen vorhanden ist.
Leaf	Speichert den Inhalt der Description von Grafikprozeduren in einer globalen Variable.
New_Image_Start	Schreibt die RTF-Ausgaben für Grafiken.
Table	Schreibt die RTF-Ausgaben für Tabellen.

In Proc Template werden Tagsets über das parent-Kommando vererbt. Allerdings kann man ein vererbtes Event nicht anpassen: Entweder übernimmt man es oder definiert es komplett selbst. Daher speichern wir die originale Event-Definition in einem neuen

Event gleichen Namens mit Suffix „_orig“, um darauf zurückgreifen zu können. Die originalen Definitionen erhalten wir so:

```
proc template;
     source tagsets.Rtf;
quit;
```

Einen Einstieg in Proc Template und Events bieten [2], [3] und [4]. Hier nur soviel: Events bestehen aus Aktionen, die bei Start oder Ende des Events ausgeführt werden. Sie haben keine Parameter, sondern kommunizieren mit der Außenwelt über globale Variablen. Wir zeigen nun Auszüge aus den umfangreichen Event-Definitionen, die Kontrollausgaben ins Log haben wir aus Platzgründen zumeist entfernt.

```
proc template;
     measured;
     define tagset tagsets.myRTF;
          parent         = tagsets.RTF;
          default_style  = "styles.myPrinterRTF";
          uniform;

          define event anchor; * Inhaltsverzeichnis ausschalten;
          end;
     end;
run;
```

Im obigen Beispiel leiten wir das Tagset myRTF aus dem Tagset RTF ab und legen den Style fest. Danach überschreiben wir das Anchor-Event mit einem leeren Event, damit die RTF-Datei keine unerwünschten Textmarken enthält.

Das Event Addfont erstellt zu Beginn der RTF-Datei eine unsichtbare Steuertabelle, die jeder Schriftart einen Index zuweist. Diesen Index verwenden RTF-Befehle zur Auswahl einer Schriftart. Deswegen speichern wir uns für diese Befehle den Index der gewünschten Schriftart in der globalen Variable llsn_id und rufen danach das originale Event Addfont auf:

```
define event addfont;
     start:
     do /if cmp(VALUE, "Arial");
          set $llsn_id LIST_INDEX;
     done;
     trigger addfont_orig start;
end;
```

Das Event Textmarke liest den Name des Objektes aus der Variablen textmarke_in und speichert den Namen der Textmarke in der Variablen textmarke_out und die Beschriftung der Tabelle in der Variablen textmarke_text. Es erwartet in der Variable textmarke_in einen Namen im folgenden Format: <Kürzel des Codeblocks z. B. RNVERT>_<Name des Ausgabe-Objekts>_<By-Variablenliste>.

```
define event textmarke;
     putlog "Event textmarke wurde ausgelöst";
     putlog "textmarke_in: " $textmarke_in;

     * Bestimmung des Codeblocknamens;
     set $codeblock_name;
     eval $codeblock_name scan($textmarke_in, 1, '_');

     * Bestimmung des Namens des Ausgabeobjekts;
     set $ausgabe_name;
     eval $ausgabe_name scan($textmarke_in, 2, '_');

     * Bestimmung des qualifizierten Namens des Ausgabeobjekts;
     set $qual_ausgabe_name;
     eval $qual_ausgabe_name catx("_", $codeblock_name,
          $ausgabe_name);

     * Bestimmung der ByGroup-Informationen (kann auch leer sein);
     set $len_codeblock_name;
     eval $len_codeblock_name length($codeblock_name);
     set $len_ausgabe_name;
     eval $len_ausgabe_name length($ausgabe_name);
     set $len_gesamt;
     eval $len_gesamt $len_codeblock_name+$len_ausgabe_name+3;
     set $textmarke_bygroup;
     eval $textmarke_bygroup substr($textmarke_in, $len_gesamt);
     eval $textmarke_bygroup putc($textmarke_bygroup,
          "$ods_byfmt.");
     putlog "textmarke_bygroup: " $textmarke_bygroup;

     * Bestimmung des Textmarkennamens;
     set $textmarke_out;
     eval $textmarke_out catx("_", $codeblock_name, $ausgabe_name,
          $textmarke_bygroup);
     putlog "textmarke_out: " $textmarke_out;

     * Bestimmung der Beschriftung der Abbildung;
     set $textmarke_text;
     eval $textmarke_text strip(putc($textmarke_out,
          "$ods_beschriftung."));
     putlog "textmarke_text: " $textmarke_text;

     * Aufräumen;
     unset $codeblock_name;
     unset $textmarke_in;
     unset $textmarke_bygroup;
     unset $ausgabe_name;
     unset $len_codeblock_name;
     unset $len_ausgabe_name;
     unset $len_gesamt;
end;
```

Mit putlog können wir Meldungen und Zwischenergebnisse ins Log schreiben, wie am Anfang des Events oder nach den Formatierungen. Globale Variablen berechnen wir mit set und eval. In qual_ausgabe_name speichern wir den Namen des Ausgabeobjekts ohne Bygroup für das Event New_Image_Start. In der Berechnung von textmarke_text und textmarke_out benutzen wir die Formate ods_byfmt und ods_beschriftung (siehe 2.2). Die Formate müssen erst bei der SAS-Ausgabe und dem Aufruf des Events bekannt sein, nicht bei der Definition des Events. Nicht verwendet werden können die Funktionen input und put, dafür aber inputc, inputn, putc und putn. Das Event Textmarke könnte auch durch eine mit PROC FCMP definierte Funktion bereitgestellt werden; bei Tests führte dies allerdings zu Abstürzen.

Das Event ByGroup setzt die Variable in_bygroup beim Start auf 1 und am Ende auf 0:

```
define event bygroup;
     start:
     unset $in_bygroup;
     set $in_bygroup;
     eval $in_bygroup 1;

     finish:
     set $in_bygroup;
     eval $in_bygroup 0;
end;
```

Das Event Table schreibt Tabellen in die RTF-Datei. Es ruft im start-Teil zu Beginn jeder by-Gruppe das Event Textmarke auf und erstellt die öffnende Textmarke. Danach beginnt der start-Teil des originalen Events Table zur Ausgabe der Tabelle. Im finish-Teil des Events Table wird der finish-Teil des originalen Events Table ausgeführt, dann werden die ausgeblendeten Textmarken _Refa<Name> und _Refb<Name> mit dem dynamischen Feld " Tabelle 3-4" und der Beschriftungen gefüllt, alle Textmarken geschlossen und der Name der Textmarke ins Dokument ausgegeben, damit der Anwender ihn kennt und verwenden kann:

```
define event table;
     start:
     do /if $bygroup_top;
          open body_tbl;
               * Bestimmung des Textmarkennames;
               set $textmarke_in;
               eval $textmarke_in strip($bygroup_top);
               do /if $in_bygroup;
                    eval $textmarke_in catx('_', $textmarke_in,
                         strip($bygroup_name));
               done;
               * Das Event textmarke definiert $textmarke_out und
                    $textmarke_text, Textmarke öffnen;
               trigger textmarke;
               put "\fs1{ \*\bkmkstart " $textmarke_out "}";
```

```
        close;
    done;
    trigger table_orig start;

    finish:
    trigger table_orig finish;
    do /if $bygroup_top;
        open body_tbl;

        * Die beiden versteckten Textmarken öffnen, das
            dynamische Feld "Tabelle 3-4" und die Beschriftung
            erstellen, alle Textmarken schließen;
        put "\trowd\trkeep\trql\trrh0\trgaph0";
        put "\pard\plain\intbl\sb10\sa10\sl-232\cf1\ql";
        put "{{\*\bkmkstart _Refa" $textmarke_out "}";
        put "{\*\bkmkstart _Refb" $textmarke_out "}";
        put "\f" $llsn_id "\fs21";
        put "Tabelle }";
        put "{";
        put "\f" $llsn_id "\fs21";
        put "\field\flddirty";
        put "{\*\fldinst";
        put "{  STYLEREF 1 \\s }";
        put "}";
        put "{\fldrslt";
        put "{99}";
        put "}";
        put "}{";
        put "\f" $llsn_id "\fs21";
        put "-}";
        put "{";
        put "\f" $llsn_id "\fs21";
        put "\field\flddirty";
        put "{\*\fldinst";
        put "{  SEQ Tabelle \\* ARABIC \\s 1 }";
        put "}";
        put "{\fldrslt";
        put "{99}";
        put "}";
        put "}";
        put "{{\*\bkmkend _Refb" $textmarke_out "}";
        put "\f" $llsn_id "\fs21 : " $textmarke_text;
        put "{\*\bkmkend _Refa" $textmarke_out "}";
        put "\cell}";
        put "\cltxlrtb\clvertalt\clcbpat8\clpadt10\clpadft3
            \clpadr10\clpadfr3\cellx8800";
        put "{\row}";

        * Dann wird die Textmarke geschlossen;
        put "{\*\bkmkend " $textmarke_out "}";
        * Und zum Schluß der Name der Textmarke ausgegeben;
        put "\trowd\trkeep\trql\trrh0\trgaph0";
```

```
          put "\pard\plain\intbl\sb10\sa10\sl-232\fs22\cf1\ql\f1{";
          put $textmarke_out;
          put "\cell}";
          put "\cltxlrtb\clvertalt\clcbpat8\clpadt10\clpadft3
               \clpadr10\clpadfr3\cellx8800";
          put "{\row}";
          close;

          * Aufräumen;
          unset $bygroup_top;
          unset $bygroup_name /if $in_bygroup;
          unset $textmarke_out;
          unset $textmarke_text;
     done;
end;
```

Wir ergänzen das Event Branch: Es speichert das content-Attribut in der Variablen bygroup_top und die aktuelle by-Gruppe in der Variablen bygroup_name.

```
define event branch;
     do /if $toc_level;
          eval $temp_toc inputn(TOC_LEVEL, "BEST");
          break /if $toc_level lt $temp_toc;
     done;
     do /if $in_bygroup;
          do /if cmp( "bycontentfolder", STYLE_ELEMENT);
               set $bygroup_name;
               eval $bygroup_name LABEL;
          done;
     done;
     do /if cmp( "contentfolder", STYLE_ELEMENT);
          set $tmp_check;
          eval $tmp_check  findc(LABEL, '_', 'ulkd');
          * Zeichen, die weder Buchstaben, Zahlen noch Unterstrich
               sind, deuten auf eine By-Gruppe hin;
          do /if $tmp_check = 0;
               set $bygroup_top;
               eval $bygroup_top LABEL;
          else;
               unset $bygroup_top;
          done;
     done;
     break /if ^$toc_data;
     open toc_tbl;
          put "{\plain\f1\b0\i0\tc\v " VALUE "  \tcf67 \tcl"
                TOC_LEVEL " }" NL;
     close;
end;
```

Das Event Leaf speichert in der Variablen grseg_top den Inhalt der Description-Option der Grafikprozeduren und ruft dann das originale Event Leaf auf. Das Event muss vor allem für die Kombination aus PROC GREPLAY und PROC GCHART angepasst werden, da PROC GCHART eine Default-Belegung der Description-Option verwendet („The Gchart procedure ….“), die wir nicht in der Ausgabe verwenden wollen. Um das Beispiel einfach zu halten, haben wir hier auf diesen Sonderfall verzichtet.

```
define event leaf;
     start:
     set $grseg_top;
     eval $grseg_top VALUE;
          * bei Verwendung von GReplay muss man hier mehr tun;
     trigger leaf_orig;

     finish:
     unset $grseg_top;
end;
```

Das Event New_Image_Start schreibt Grafiken in die RTF-Datei. Es ruft im start-Teil zu Beginn jeder by-Gruppe das Event Textmarke auf und erstellt die öffnende Textmarke. Dann bettet es die Grafik ins RTF. Wir fügen zwei Befehle ein, um die Grafik zurechtzuschneiden. Im finish-Teil werden die ausgeblendeten Textmarken _Refa<Name> und _Refb<Name> mit dem dynamischen Feld "Abbildung 2-3" und der Beschriftungen gefüllt, alle Textmarken geschlossen und der Name der Textmarke ins Dokument ausgegeben, damit der Anwender ihn kennt und verwenden kann:

```
define event new_image_start;
     start:
     set $stream_name VALUE;
     open $stream_name;
     do /if $grseg_top;
          set $textmarke_in;
          eval $textmarke_in $grseg_top;
          trigger textmarke;
          put "{\*\bkmkstart " $textmarke_out "}";
     done;

     put "{\*\shppict{\pict\pngblip" /when cmp( LABEL, "PNG");
     put "{\*\shppict{\pict\jpegblip" /when cmp( LABEL, "JPEG");
     put "{\*\shppict{\pict\emfblip" /when cmp( LABEL, "EMF");
     put "{\*\shppict{\pict\jpegblip" /when cmp( LABEL, "JFIF");
     put "{\*\shppict{\pict\jpegblip" /when cmp( LABEL, "JPG");
     put "\picwgoal" OUTPUTWIDTH "\pichgoal" OUTPUTHEIGHT " " NL;

     * Folgender Befehl schneidet oben und unten etwas vom Bild ab.;
     do /if $grseg_top;
          set $grseg_abschneiden_oben;
          eval $grseg_abschneiden_oben
               coalesce(inputn($qual_ausgabe_name, "ods_oben."), 0);
```

```
     do /if $grseg_abschneiden_oben;
          put "\piccropt";
          put $grseg_abschneiden_oben;
     done;
     set $grseg_abschneiden_unten;
     eval $grseg_abschneiden_unten
          coalesce(inputn($qual_ausgabe_name, "ods_unten."), 0);
     do /if $grseg_abschneiden_unten;
          put "\piccropb";
          put $grseg_abschneiden_unten NL;
     done;
done;

finish:
put "}}" NL;
* Die beiden versteckten Textmarken öffnen, das
     dynamische Feld für "Abbildung 2-3" und die Beschriftung
     erstellen, alle Textmarken schließen;
do /if $grseg_top;
     put "{\par}\pard\plain";
     put "{{\*\bkmkstart _Refa" $textmarke_out "}";
     put "{\*\bkmkstart _Refb" $textmarke_out "}";
     put "\f" $llsn_id "\fs21";
     put "Abbildung }";
     put "{";
     put "\f" $llsn_id "\fs21";
     put "\field\flddirty";
     put "{\*\fldinst";
     put "{  STYLEREF 1 \\s }";
     put "}";
     put "{\fldrslt";
     put "{99}";
     put "}";
     put "}{";
     put "\f" $llsn_id "\fs21";
     put " -";
     put "}";
     put "{";
     put "\f" $llsn_id "\fs21";
     put "\field\flddirty";
     put "{\*\fldinst";
     put "{  SEQ Abbildung \\* ARABIC \\s 1 }";
     put "}";
     put "{\fldrslt";
     put "{99}";
     put "}";
     put "}";
     put "{{\*\bkmkend _Refb" $textmarke_out "}";
     put "\f" $llsn_id "\fs21";
     put ": " $textmarke_text;
     put "{\*\bkmkend _Refa" $textmarke_out "}";
     put "\par}";
```

```
          * Dann wird die Textmarke geschlossen;
          put "{{\*\bkmkend " $textmarke_out "}";
          put "\par}" NL;
          * Und den Name der Textmarke ausgegeben;
          put "{\ql " $textmarke_out " \par}";
          unset $grseg_top;
          unset $textmarke_out;
          unset $textmarke_text;
     done;
     close;
end;
```

Wir stutzen die Grafik mit den Informaten ods_oben und ods_unten, um Titel und Fußnote der Grafik zu entfernen (siehe 2.4). Die Informate weisen einem Grafiknamen eine natürliche Zahl zu, die angibt, wieviel twip[1] abgeschnitten werden. Werte kleiner als 1 führen zu einem Fehler, doch 1 führt zu einer kaum sichtbaren Änderung.

2.4 Zusammenspiel mit ODS PDF

Parallel zur Ausgabe als RTF erzeugen wir auch ein PDF, um sämtliche Ergebnisse revisionssicher zu dokumentieren. Dabei sind die Ziele unterschiedlich: Das RTF dient als Brücke zu Word und benötigt nicht viel Text, da das Ergebnis im Bericht kommentiert wird; im PDF steht das Ergebnis allein und benötigt Titel sowie Fußnoten zum besseren Verständnis. Wir wollen das SAS-Programm nicht für die einzelnen Ausgabekanäle duplizieren und unterstützen daher mit folgenden Mitteln die unterschiedlichen Ziele:

- Grafiken: Wir schneiden Titel und Fußnoten im RTF ab (siehe Event New_Image_Start in 2.3).
- Tabellen: Für das PDF erzeugen wir in PROC REPORT mittels line-Statements Titel und Fußnoten. Diese verstecken wir im RTF mit Zeilengröße 1pt und weißer Schrift. So bleiben diese unauffällig und können später hervorgehoben oder gelöscht werden.

Für die Tabellen erstellen wir dazu einen Style für das RTF aus dem SAS-Style printer:

```
proc template;
     define style myPrinterRTF;
          parent = styles.printer;
          replace fonts /
               'TitleFont'    = ("Lucida Bright",18pt)
               'docFont'      = ("Lucida Sans Narrow",10.5pt)
               […]
          ;
          class TabellenUeberschrift
               "Dieser Style blendet die Tabellenueberschrift aus" /
               color     = white
               font_size = 1pt;
```

[1] twentieth of an inch point

```
          ;
          class TabellenFussNote
               "Dieser Style blendet Tabellenbeschreibungen aus" /
               color     = white
               font_size = 1pt;
          ;
     end;
run;
```

Wählt man als Schriftgröße genau 10 Punkt, wird die Schriftgröße nicht im RTF vermerkt, was beim Übertragen in den Bericht zu Problemen führen kann. Daher wählen wir 10,5. Dies ist unproblematisch, da die Tabellen in Word zumeist kleiner eingefügt werden als in ein paralleles PDF. Die beiden Klassen machen unerwünschte Überschriften und Fußnoten unsichtbar. Sie werden so angewendet:

```
proc report data=mydata  contents="RPT_UEB";
     [...];

     compute before _page_ / style = TabellenUeberschrift;
          line 'Titel';
     endcomp;

     compute after / style = TabellenFussNote;
          line 'Fußnote';
     endcomp;
run;
```

3 Einschränkungen

Einschränkungen sind:

- In Proc Report sind inline-Formatierungen in den line-Statements wirkungslos: `"~S={font_weight = bold}"`.
- Beim Einfügen von Tabellen in Word entsteht eine Leerzeile über der Tabelle.
- Um Vektorgrafiken im RTF zu erzeugen, legen wir in den Grafikoptionen fest: device=SASEMF. In der HTML-Ausgabe führt dies zu einer Warnung.
- Verwendung von PROC GREPLAY: Nur replay verwendet das Description-Attribut der originalen Grafik, treplay dagegen nicht.

4 Ausblick

Wir haben das Tagset etwas vereinfacht, um die Idee zu erläutern, es ist dennoch lauffähig. Die Beschriftungen einzelner Ergebnisse geben wir zentral vor und kombinieren sie mit den Formaten dynamisch zur Beschriftung. Das Einbinden in Word erleichtern wir unseren Anwendern weiter mit einem VBA-Makro, das aus den RTF-Dateien in einem Verzeichnis ausliest, welche Textmarken darin vorhanden sind und deren Beschriftungen anzeigt. Zusätzlich hilft es beim Einbinden und Aktualisieren der Objekte.

Literatur

[1] Microsoft Corporation: Rich Text Format (RTF) Specification, Version 1.9.1. Microsoft Corporation, 2008. Verwendete Version heruntergeladen am 15.02.2017 von http://www.microsoft.com/.

[2] L. Haworth: “PROC TEMPLATE: The Basics”, Paper 112 aus Proceedings of the Thirty-first Annual SAS® Users Group International Conference, SAS Institute Inc., Cary, North Carolina 2006.

[3] C. Zender: “Creating a Tagset Template for The SAS® XML Libname Engine”, Paper TT-24 aus Proceedings of the Pharmaceutical Industry SAS® Users Group Conference, SAS Institute Inc., Cary, North Carolina 2006.

[4] SAS Institute Inc.: SAS® 9.3 Output Delivery System: User's Guide, Second Edition. SAS Institute Inc., Cary, North Carolina 2012.