Paper 019-31

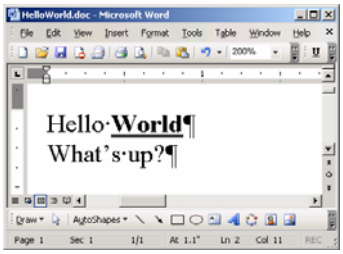
Reading Microsoft Word XML files with SAS®

Larry Hoyle Policy Research Institute, The University of Kansas, Lawrence, KS

ABSTRACT

In 2005 Microsoft announced that the new default format for documents created in Microsoft Office will be XML-based. The ability of SAS to read XML offers a convenient method for extracting structured information from Microsoft Word documents. This paper examines three scenarios where information from a Word document is read into SAS datasets: extracting text along with associated properties (styles and attributes), extracting all data from tables, and extracting coordinates of objects in drawings.

MICROSOFT WORD XML



The example documents in this paper were created in Microsoft Word 2003. When saved as XML and viewed in raw form the "Hello World" document appears as at right.

XML elements have a hierarchical structure. Note that the display of some of the elements such as **w:fonts** have been collapsed in this view for clarity's sake. The **w:body** element is shown fully expanded. The word "Hello" can be seen to be embedded in a **w:t** element within a **w:r** element within a **w:p** element within a **wx:sect** element within the **w:body** element within the **w:wordDocument** element. The **w:t** element is the **t** element defined in the namespace prefix **w** found at: xmlns:w="http://schemas.microsoft.com/office/word/2003/wordml"

The **t** elements contain fragments of document text.

The **w:r** element, which defines a "run", exists to allow text to be assigned properties. In this case the word "World" is assigned the **w:b** (bold) and **w:u** properties (underline) properties as signified by the presence of these elements in the run element. The **w:u** element itself has an attribute **val** which has the value of "single" – it is a single underline.

Properties can also be applied to text through styles. Styles can be specified either to a run or to a paragraph. Properties are assigned to a paragraph in the **pPr** element.

Documentation for the Microsoft Office XML schemas can be found at the Web site listed in the references below.

Scenario 1, extracting text and attributes

Suppose you have a document in which passages of text are marked with attributes that categorize them. You might want to use SAS Text Miner on the documents to categorize the text statistically and then compare the manual categorizations to the statistically generated ones.

The manual coder marked:

comments that seemed negative in red
comments that seemed positive in green
comments that dealt with the way the person was treated with the style "Treated"
comments that dealt with waiting time with the style "Waiting"

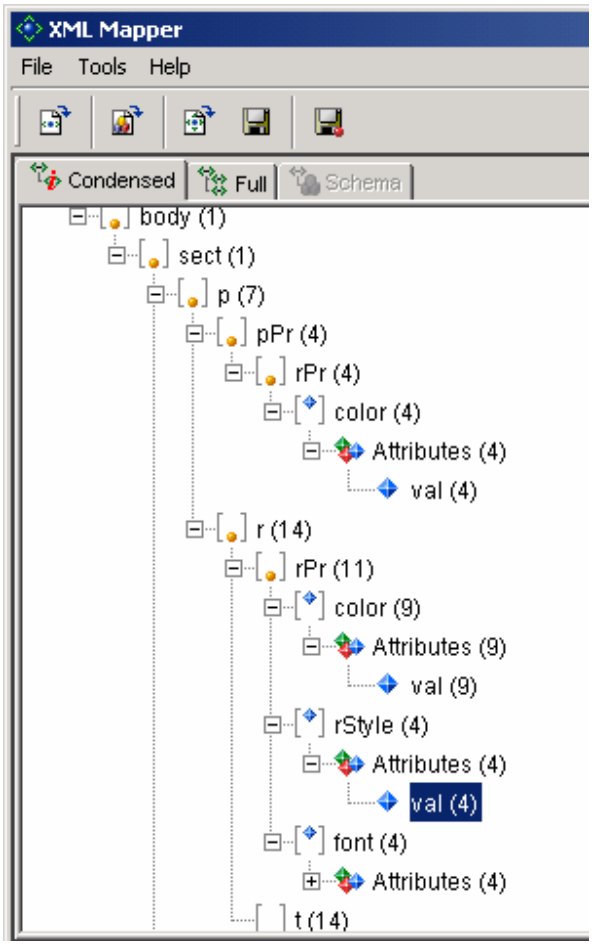
Note that this scheme gives us two independent dimensions for any particular word in the text, one dealing with affect – positive or negative, and the other dealing with a categorization of the issue involved – waiting time or treatment. A given paragraph might have segments of text coded with multiple possible combinations.

Here is a sample coded text file. "Treated" style appears in this font . "Waiting" style appears in this font.

Line 1 is red
Line 2 is green except for the last word
Line 3 is black
Line 4 is black
Line 5 is red
Line 6 is green

I have never been so humiliated in my life. That was very rude treatment.
What a pleasant experience. Your staff was both quick and pleasant.
It took about the time I expected to reach someone.
I have nothing to say. The sky is blue and the sea is green.
You are the worst organization in the world.
I love you guys.

The outline in this figure, taken from SAS XML Mapper shows the XML elements needed to read the text and coding into SAS. Within each section of the document there are paragraph elements **p,** which in turn contain run elements **r**. The run elements contain text elements **t**, which contain fragments of the document's text. Color attributes appear in run property elements **rPr**, in turn within both paragraph property elements **pPr** and run elements **r.** The run property elements within the paragraph property elements apply only to the paragraph mark and not to the text within the paragraph.



In our sample document, style properties **rStyle** appear only in the run properties.
A SAS dataset to contain this text and maintain the hierarchical structure might have a record for each text fragment with variables for the section, paragraph, run and text fragment numbers, as well as the text itself, the paragraph and run color and the run style.

To do this SAS needs an XML Map file. The XML Map file can either be typed manually or created using XML Mapper. The XML Map file describes how to read any XML file with the same structure.
An important notion for this description is the XML path. Note in the outline at right that there are **rPr** elements embedded within **pPr** elements and there are **rPr** elements embedded within **r** elements. The former would have the path:
/w:wordDocument/w:body/wx:sect/w:p/w:pPr
and the latter would have the path:
/w:wordDocument/w:body/wx:sect/w:p/w:r/w:rPr.
The XML Map file specifies the XML path which becomes rows in the SAS dataset with the **TABLE-PATH** element.. It specifies the XML paths which become columns in the SAS dataset with the **PATH** element within the **COLUMN** element.. It can also specify to place an ordinal number into the dataset instead of the content via the **INCREMENT-PATH** element, This is how paragraph numbers, etc. can be generated. See, for example the

```
<COLUMN name="pNum" ordinal="YES" retain="YES">
```

element , which creates paragraph numbers, in the XML MAP below.
The ability to generate these ordinals would be especially important in reading Word XML documents that were generated by an application other than Word. The **wx:sect** element is really an optional annotation. Sections are actually delineated by a **w:sectPr** element appearing in the last paragraph of each section.

XML Map File to Read the Sample Text

```
<?xml version="1.0" encoding="windows-1252"?>
<!-- ############################################################ -->
<!-- 2005-07-30T11:12:26 -->
<!-- SAS XML Libname Engine Map -->
<!-- Generated by XML Mapper, 9.1.0300.20040709.2028 -->
<!-- ############################################################ -->
<!-- ###  Validation report                                   ### -->
<!-- ############################################################ -->
<!-- RETAIN option is not set for column (RStyleval) in table (t), though XPath comparisions indicate it should be. -->
<!-- Map validation completed successfully. -->
<!-- ############################################################ -->
<SXLEMAP xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="textSample" version="1.2"
xsi:noNamespaceSchemaLocation="http://www.sas.com/xml/schema/sxle12.xsd">
    <!-- ############################################################ -->
<TABLE name="t">
<TABLE-PATH syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:t</TABLE-PATH>
<COLUMN name="sectNum" ordinal="YES" retain="YES">
   <INCREMENT-PATH beginend="BEGIN" syntax="XPath">/w:wordDocument/w:body/wx:sect</INCREMENT-PATH>
   <TYPE>numeric</TYPE>
   <DATATYPE>integer</DATATYPE>
</COLUMN>
<COLUMN name="pNum" ordinal="YES" retain="YES">
    <INCREMENT-PATH beginend="BEGIN" syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p</INCREMENT-PATH>
    <TYPE>numeric</TYPE>
    <DATATYPE>integer</DATATYPE>
</COLUMN>
<COLUMN name="rNum" ordinal="YES" retain="YES">
    <INCREMENT-PATH beginend="BEGIN" syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r</INCREMENT-PATH>
    <TYPE>numeric</TYPE>
    <DATATYPE>integer</DATATYPE>
</COLUMN>
<COLUMN name="tNum" ordinal="YES" retain="YES">
  INCREMENT-PATH beginend="BEGIN" syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:t</INCREMENT-PATH>
  TYPE>numeric</TYPE>
 <DATATYPE>integer</DATATYPE>
</COLUMN>
<COLUMN name="t">
    <PATH syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:t</PATH>
    <TYPE>character</TYPE>
    <DATATYPE>string</DATATYPE>
    <LENGTH>60</LENGTH>
</COLUMN>
<COLUMN name="PColorVal" retain="YES">
    <PATH syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:pPr/w:rPr/w:color/@val</PATH>
    <TYPE>character</TYPE>
    <DATATYPE>string</DATATYPE>
    <LENGTH>6</LENGTH>
</COLUMN>
<COLUMN name="RColorVal" retain="YES">
    <PATH syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:rPr/w:color/@val</PATH>
    <TYPE>character</TYPE>
    <DATATYPE>string</DATATYPE>
    <LENGTH>6</LENGTH>
</COLUMN>
<COLUMN name="RStyleval">
    <PATH syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:rPr/w:rStyle/@val</PATH>
    <TYPE>character</TYPE>
    <DATATYPE>string</DATATYPE>
    <LENGTH>11</LENGTH>
</COLUMN>
</TABLE>
</SXLEMAP>
```

The data as read in to SAS

XML Mapper can also generate a sample SAS program which sets up the libname that uses the XML Map file to read the data in to SAS.

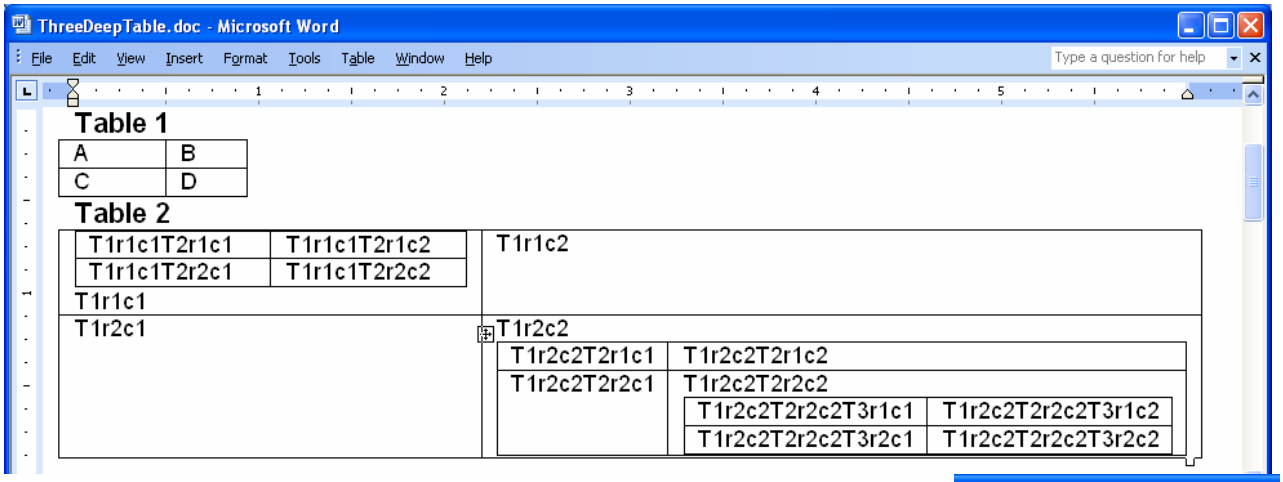
The SAS program is listed below and a view of the dataset appears to the right.

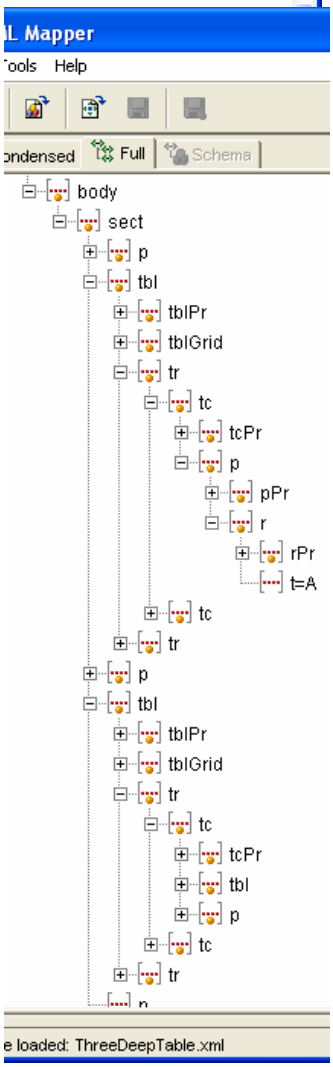
VIEWTABLE: Textsamp.T

	sectNum	pNum	rNum	tNum	t	PColorVal	RColorVal	RStyleval
1	1	1	1	1	I have never been so humiliated in my life. That was very	FF0000	FF0000	
2	1	1	2	2	rude treatment		FF0000	TreatedChar
3	1	1	3	3	.		FF0000	
4	1	2	4	4	What a pleasant experience. Your staff was both	008000	008000	
5	1	2	5	5	quick		008000	waitingChar
6	1	2	6	6	and		008000	
7	1	2	7	7	pleasant			TreatedChar
8	1	2	8	8	.		008000	
9	1	3	9	9	It			
10	1	3	10	10	took about the time I expected			waitingChar
11	1	3	11	11	to reach someone.			
12	1	4	12	12	I have nothing to say. The sky is blue and the sea is green.			
13	1	5	13	13	You are the worst organization in the world.	FF0000	FF0000	
14	1	6	14	14	I love you guys.	008000	008000	

```
/*******************************************************************************
 *  Generated by XML Mapper, 9.1.0300.20040709.2028
 *******************************************************************************/

/*
 *  ENVIRONMENT
 */
filename  textSamp 'C:\ddrive\projects\sugs\sugi31\WordXML\sampleDocuments\textSample.xml';
filename  SXLEMAP 'C:\ddrive\projects\sugs\sugi31\WordXML\XMLmaps\sampleText.map';
libname   textSamp xml xmlmap=SXLEMAP access=READONLY;

/*
 *  CATALOG
 */

proc datasets lib=textSamp; run;

/*
 *  SAMPLE USAGE
 */

title 'Table t';
proc contents data=textSamp.t varnum; run;
proc print data=textSamp.t; run;
```

Scenario 2, extracting data from tables





It is also possible to create an XML Map file which can read all of the data in tables in a Word Document. The sample document shown above has two tables. The XML hierarchy of interest is shown at right. Data in tables is in text fragments **t** within runs **r** within paragraphs **p** within table columns **tc** within table rows **tr** within tables **tbl.** Tables may be nested within table column elements.

The Xpaths in the first example were all *absolute* paths, that is they began at the root element of the XML document. Absolute paths would not allow us to capture data from tables nested arbitrarily deeply. The Xpath:

/w:wordDocument/w:body/wx:sect/w:tbl/w:tr/w:tc/w:p/w:r/**w:t**

would only capture the data in non-nested cells like “C: and “T1r1c1” in the sample table. An alternative would be to use relative paths. The Xpath

w:tc/w:p/w:r/w:t

selects only text that occurs within a table cell, but that table cell may itself be in nested tables. The only remaining trick, then, is to capture the nesting structure. This can be done by generating ordinals both for the beginning of elements and the end of elements. This INCREMENT-PATH generates an ordinal for the beginning of table elements

<INCREMENT-PATH beginend="BEGIN" syntax="XPath">w:tbl</INCREMENT-PATH>

This one generates increments at the end of table elements

<INCREMENT-PATH beginend="End" syntax="XPath">w:tbl</INCREMENT-PATH>

The combination allows each piece of text to be placed in its proper place in the nested structure.

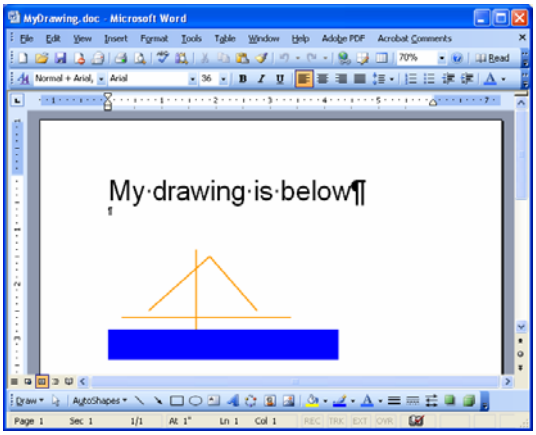
	tblBegin	tblEnd	trBegin	trEnd	tcBegin	tcEnd	pNum	rNum	tNum	t
1	1	.	1	.	1	.	2	2	2	A
2	1	.	1	.	2	1	3	3	3	B
3	1	.	2	1	3	2	4	4	4	C
4	1	.	2	1	4	3	5	5	5	D
5	3	1	4	2	6	4	7	7	7	T1r1c1T2r1c1
6	3	1	4	2	7	5	8	8	8	T1r1c1T2r1c2
7	3	1	5	3	8	6	9	9	9	T1r1c1T2r2c1
8	3	1	5	3	9	7	10	10	10	T1r1c1T2r2c2
9	3	2	5	4	9	8	11	11	11	T1r1c1
10	3	2	5	4	10	9	12	12	12	T1r1c2
11	3	2	6	5	11	10	13	13	13	T1r2c1
12	3	2	6	5	12	11	14	14	14	T1r2c2
13	4	2	7	5	13	11	15	15	15	T1r2c2T2r1c1
14	4	2	7	5	14	12	16	16	16	T1r2c2T2r1c2
15	4	2	8	6	15	13	17	17	17	T1r2c2T2r2c1
16	4	2	8	6	16	14	18	18	18	T1r2c2T2r2c2
17	5	2	9	6	17	14	19	19	19	T1r2c2T2r2c2T3r1c1
18	5	2	9	6	18	15	20	20	20	T1r2c2T2r2c2T3r1c2
19	5	2	10	7	19	16	21	21	21	T1r2c2T2r2c2T3r2c1
20	5	2	10	7	20	17	22	22	22	T1r2c2T2r2c2T3r2c2

XML Map to Read Tables

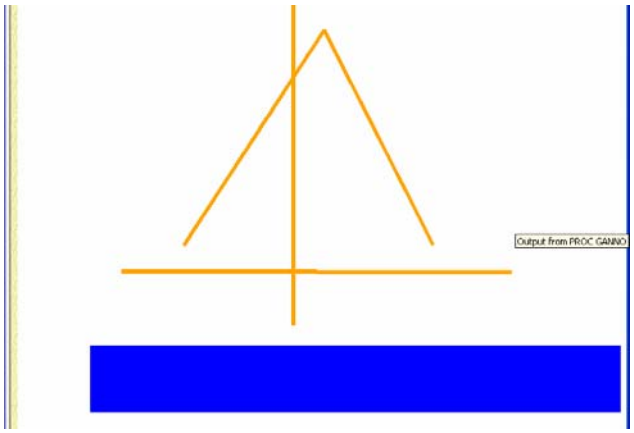
```
<?xml version="1.0" encoding="windows-1252"?>

<SXLEMAP xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Tables" version="1.2"
xsi:noNamespaceSchemaLocation="http://www.sas.com/xml/schema/sxle12.xsd">

    <TABLE name="t">
        <TABLE-PATH syntax="XPath">w:tc/w:p/w:r/w:t</TABLE-PATH>

        <COLUMN name="tblBegin" ordinal="YES" retain="YES">
            <INCREMENT-PATH beginend="BEGIN" syntax="XPath">w:tbl</INCREMENT-PATH>
            <TYPE>numeric</TYPE>
            <DATATYPE>integer</DATATYPE>
        </COLUMN>
        <COLUMN name="tblEnd" ordinal="YES" retain="YES">
            <INCREMENT-PATH beginend="End" syntax="XPath">w:tbl</INCREMENT-PATH>
            <TYPE>numeric</TYPE>
            <DATATYPE>integer</DATATYPE>
        </COLUMN>
        <COLUMN name="trBegin" ordinal="YES" retain="YES">
            <INCREMENT-PATH beginend="BEGIN" syntax="XPath">w:tr</INCREMENT-PATH>
            <TYPE>numeric</TYPE>
            <DATATYPE>integer</DATATYPE>
        </COLUMN>
        <COLUMN name="trEnd" ordinal="YES" retain="YES">
            <INCREMENT-PATH beginend="End" syntax="XPath">w:tr</INCREMENT-PATH>
            <TYPE>numeric</TYPE>
            <DATATYPE>integer</DATATYPE>
        </COLUMN>
        <COLUMN name="tcBegin" ordinal="YES" retain="YES">
            <INCREMENT-PATH beginend="Begin" syntax="XPath">w:tc</INCREMENT-PATH>
            <TYPE>numeric</TYPE>
            <DATATYPE>integer</DATATYPE>
        </COLUMN>
        <COLUMN name="tcEnd" ordinal="YES" retain="YES">
            <INCREMENT-PATH beginend="End" syntax="XPath">w:tc</INCREMENT-PATH>
            <TYPE>numeric</TYPE>
            <DATATYPE>integer</DATATYPE>
        </COLUMN>
        <COLUMN name="pNum" ordinal="YES" retain="YES">
            <INCREMENT-PATH beginend="BEGIN" syntax="XPath">w:p</INCREMENT-PATH>
            <TYPE>numeric</TYPE>
            <DATATYPE>integer</DATATYPE>
        </COLUMN>
        <COLUMN name="rNum" ordinal="YES" retain="YES">
            <INCREMENT-PATH beginend="BEGIN" syntax="XPath">w:r</INCREMENT-PATH>
            <TYPE>numeric</TYPE>
            <DATATYPE>integer</DATATYPE>
        </COLUMN>
        <COLUMN name="tNum" ordinal="YES" retain="YES">
            <INCREMENT-PATH beginend="BEGIN" syntax="XPath">w:t</INCREMENT-PATH>
            <TYPE>numeric</TYPE>
            <DATATYPE>integer</DATATYPE>
        </COLUMN>
        <COLUMN name="t">
            <PATH syntax="XPath">w:tc/w:p/w:r/w:t</PATH>
            <TYPE>character</TYPE>
            <DATATYPE>string</DATATYPE>
            <LENGTH>30</LENGTH>
        </COLUMN>
    </TABLE>
</SXLEMAP>
```

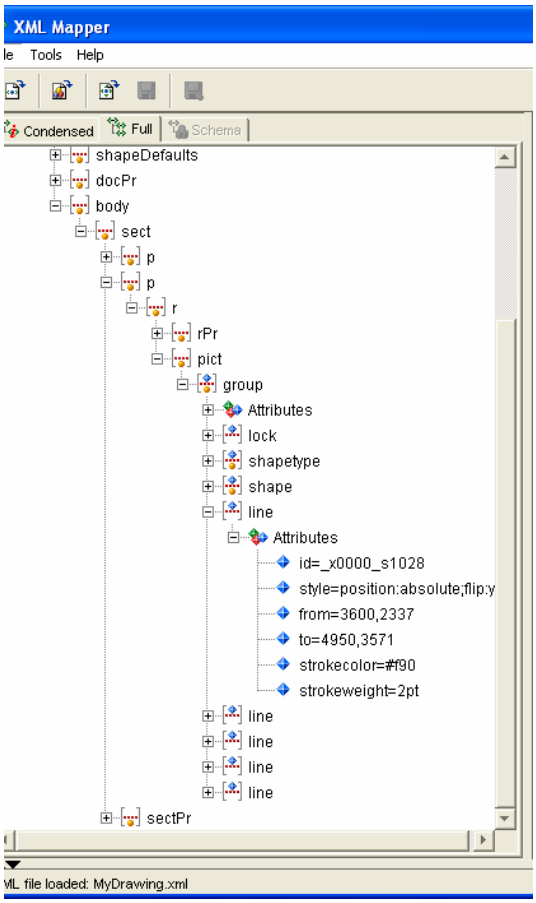
Scenario 3, extracting drawing object parameters (digitizing)

The Microsoft Word document shown here has an embedded drawing made up of straight line segments. It could have other graphical objects – rectangles, squiggles, arrows, ovals and so on, but for simplicity's sake we'll just look a figure with lines. When saved as an XML file, the lines are elements inside a **group** inside a run as seen in the screen image taken from SAS XML Mapper. An XMLMap file defining rows at the level of the line object and with columns for **id, style from, to, strokecolor, strokeweight**, and with ordinals generated for **line** and **pict** would allow one to re-plot the figure with a SAS Annotate dataset. The SAS code for that is shown below.



Here is the figure as plotted in SAS:





SAS code for reading and plotting lines

```
 /*****************************************
 Read and plot the lines from the first picture
in a Word document

*****************************************/


filename  MyDrawin 'C:\MyDrawing.xml';
filename  SXLEMAP 'C:\MyDrawing.map';
libname   MyDrawin xml xmlmap=SXLEMAP
access=READONLY;

proc format;
invalue $wcolor
'#f90' = 'orange'
'blue' = 'blue';
run;
```

```
data MyAnno;
 set Mydrawin.line;
 where pictNum=1;
 length function style color $ 8;
 retain xsys ysys hsys '2' line 1;

 color=input(strokecolor,$wcolor.);

                                          /*  native units are 1200/inch   */
                                          /* stroke is in points (72/inch) */
 strokePattern = prxparse('/(\d+)(\D+)/');
 if prxmatch(strokePattern, strokeweight) then do;
   size=int(167*input(prxposn(strokePattern,1,strokeweight),10.));
   units=trim(prxposn(strokePattern,2,strokeweight))||' * 16.667';
 end;
 else put 'stroke weight not matched';
 drop xypattern strokePattern ;
  xyPattern = prxparse('/(\d+),(\d+)/');

 if prxmatch(xyPattern, from) then do;
   function='move';
                                          /* NOTE: the style element may indicate
*/
                                          /* that the form and to y values are */
                                          /* flipped*/
   x= input(PRXPOSN (xyPattern, 1,  from),10.);
   if prxmatch('/flip:y/',style) then
     y= -1* input(PRXPOSN (xyPattern, 2,  to),10.);
   else
     y= -1* input(PRXPOSN (xyPattern, 2,  from),10.);
   output;


   if prxmatch(xyPattern, to) then do;
     function='draw';
     x= input(PRXPOSN (xyPattern, 1,  to),10.);
   if prxmatch('/flip:y/',style) then
     y= -1* input(PRXPOSN (xyPattern, 2,  from),10.);
   else
     y= -1* input(PRXPOSN (xyPattern, 2,  to),10.);       output;
   end;

  end;    /* prxmatch on from  */

run;



proc ganno annotate=Myanno datasys;
run;
```

XML MAP file for reading lines

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- ############################################################ -->
<!-- 2005-07-29T16:12:30 -->
<!-- SAS XML Libname Engine Map -->
<!-- Generated by XML Mapper, 9.1.0300.20040709.2028 -->
<!-- ############################################################ -->
<!-- ###  Validation report                                   ### -->
<!-- ############################################################ -->
<!-- Map validation completed successfully. -->
<!-- ############################################################ -->
<SXLEMAP xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="WordDrawing" version="1.2"
xsi:noNamespaceSchemaLocation="http://www.sas.com/xml/schema/sxle12.xsd">

    <!-- ############################################################ -->
<TABLE name="line">
<TABLE-PATH syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:pict/v:group/v:line</TABLE-PATH>

<COLUMN name="id">
    <PATH syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:pict/v:group/v:line/@id</PATH>
    <TYPE>character</TYPE>
    <DATATYPE>string</DATATYPE>
    <LENGTH>12</LENGTH>
</COLUMN>

<COLUMN name="style">
    <PATH syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:pict/v:group/v:line/@style</PATH>
    <TYPE>character</TYPE>
    <DATATYPE>string</DATATYPE>
    <LENGTH>24</LENGTH>
</COLUMN>

<COLUMN name="from">
    <PATH syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:pict/v:group/v:line/@from</PATH>
    <TYPE>character</TYPE>
    <DATATYPE>string</DATATYPE>
    <LENGTH>9</LENGTH>
</COLUMN>

<COLUMN name="to">
    <PATH syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:pict/v:group/v:line/@to</PATH>
    <TYPE>character</TYPE>
    <DATATYPE>string</DATATYPE>
    <LENGTH>9</LENGTH>
</COLUMN>

<COLUMN name="strokecolor">
    <PATH syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:pict/v:group/v:line/@strokecolor</PATH>
    <TYPE>character</TYPE>
    <DATATYPE>string</DATATYPE>
    <LENGTH>4</LENGTH>
</COLUMN>

<COLUMN name="strokeweight">
    <PATH syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:pict/v:group/v:line/@strokeweight</PATH>
    <TYPE>character</TYPE>
    <DATATYPE>string</DATATYPE>
    <LENGTH>4</LENGTH>
</COLUMN>
```

```
<COLUMN name="lineNum" ordinal="YES" retain="YES">
 <INCREMENT-PATH beginend="BEGIN"
              syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:pict/v:group/v:line</INCREMENT-PATH>
 <TYPE>numeric</TYPE>
 <DATATYPE>integer</DATATYPE>
</COLUMN>

<COLUMN name="pictNum" ordinal="YES" retain="YES">
 <INCREMENT-PATH beginend="BEGIN" syntax="XPath">/w:wordDocument/w:body/wx:sect/w:p/w:r/w:pict</INCREMENT-PATH>
 <TYPE>numeric</TYPE>
 <DATATYPE>integer</DATATYPE>
</COLUMN>

</TABLE>

</SXLEMAP>
```

OTHER OFFICE XML

Microsoft Excel has its own XML document type but the techniques described in this paper allow for the creation of generic XMLMap files to extract data and attributes from Excel spreadsheets. For an example see the XMLMap file from the DelGobbo SUGI 30 paper.
Microsoft Access, on the other hand creates a separate document type for each table or set of tables exported to XML. Exports of single tables can be read with the native format for the XML libname engine, so no XMLMap file is needed. Exports of tables with relationships can, however, produce a nested structure that requires a unique XMLMap. Access will produce an XML schema file which can be used in XMLMapper to create the XMLMap file.

REFERENCES

DelGobbo, Vincent. 2005. "Moving Data and Analytical Results between SAS® and Microsoft Office " *Proceedings of the Thirtieth Annual SAS User Group Conference*, Philadelphia, PA, 136-30.

Microsoft, Office 2003 XML Reference Schemas http://www.microsoft.com/office/xml/default.mspx
Lenz, Evan, Mary McRae, and Simon St. Laurent. 2004 *Office 2003 XML* Sebastopol, CA: O'Reilly Media Inc.

THE SAS PROGRAM AND XML MAP FILE

The SAS programs, XML files, XML Maps will be available on the Web at http://www.ku.edu/pri/ksdata/sashttp/sugi31.
Contact the author at:

Larry Hoyle
Policy Research Institute
University of Kansas, Blake Hall
1541 Lilac Lane
Lawrence, KS 66044-3177
Email: LarryHoyle@ku.edu