

The Version 7 Output Delivery System

Information Products Staff
SAS Institute Inc., Cary, NC

Abstract

This paper discusses advances made in the output generated by all SAS® Software procedures. New features provided by the Output Delivery System [ODS] include HTML generation, output datasets from any printed output and a persistable output document that can be re-organized and re-played long after the Proc that created it has completed. ODS also produces all legacy line-mode printing. The talk establishes the base line ODS features to be delivered in Version 7, as well as a roadmap to future enhancements.

History

SAS Output has been criticized for its Green-Bar characteristics. [Green-Bar being a generic term for "traditional" computer paper mounted in a chain printer – perforations along the sides, 66 lines deep, 132 characters wide, horizontal green stripes to aid readability... The stuff legends are made of.]

Until Version 7, SAS procedures that produce printed output (i.e. destined for the SAS Listing File) have had this "page of paper" as their model when generating output. Headings are centered within the LS= values. PS= determines how many lines will print on a page. Each procedure was responsible for formatting its own output, and this is often done one line at a time, moving characters into the "print buffer" of the program.

PROC PRINTTO has had a 20+ year reign as the "solution" to capturing output that is printed by one SAS procedure, but is required as input for some other procedure. Many SAS procedures produce output datasets with an OUTPUT Statement or an OUTxx= option, but invariably there is some printed data point that the programmer wishes he could "capture" that is not available on the output data set.

SAS Users Want Better Output

Our customers want a lot more from SAS Output. Features like:

- Ability to get all printed numbers into datasets with no loss of precision

- Ability to select and exclude different pieces of output, and to combine and rearrange tables.
- Ability to format with better display technologies like HTML, SGML, Postscript and PDF.
- Ability to embed [or simply send] SAS Output directly into word processing applications.

This would be a tall order if each procedure were to retain the responsibility for formatting its own output. Each procedure would have to grow the code to do all of the marvelous things mentioned above – clearly we needed a better way.

Steps Toward Better Output

Overhauling more than 100 procedures to treat output more comprehensively was no small task. We started in Version 6 with a prototype of ODS [Olinger+Ray, SUGI92] that introduces the concept of output as self-describing objects that contain both data and "instructions" for formatting that data. A SAS procedure with complicated output may create several such objects, which are often naturally arranged in a hierarchy. Consider this GLM code fragment that models two variables, x and y:

```
data a;
  do a = 1 to 4;
    do b = 1 to 4;
      do c = 1 to 4;
        n = int(6*ranuni(1));
        do i = 1 to n;
          y = a + b
            + rannor(1);
          x = a * b
            / rannor(3);
          output;
        end; end; end; end;

proc glm data=a;
  class a b c;
  model x y = a|b|c
    / ss1 ;
run;
```

The Output from this code arranges into a hierarchy quite comfortably. *Italics* are used to show actual

output objects. **Bold** is used to show "directory" objects created to make a useful structure.

In this specific GLM example, a **Data** directory is created to summarize the values found in the input data set, and an **Analysis of Variance** directory is created for the results of the analysis. A separate directory is created for each variable being modeled.

GLM

Data

Class Levels

Number of Observations

Analysis of Variance

Dependent Variable X

Overall ANOVA

Fit Statistics

Type I Model ANOVA

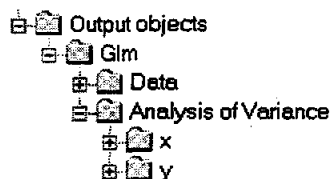
Dependent Variable Y

Overall ANOVA

Fit Statistics

Type I Model ANOVA

When run through ODS this same hierarchy is displayed using a tree control in the new SAS explorer window:



Selecting one of the directories (Y for example) shows the available tables in that directory in the right hand pane of the explorer.

Contents of y		
Name	Size	Type
Overall ANOVA		Output
Fit Statistics		Output
Type I Model ANOVA		Output
Type II Model ANOVA		Output
Type III Model ANOVA		Output
Type IV Model ANOVA		Output

ODS removes the responsibility for formatting output from the procedure completely. The procedure supplies the raw data [along with the name of the formatting instructions] to ODS, and ODS takes care of the details.

ODS Details

All procedures that produce printed output have been converted to send that output through ODS. The SAS programmer interacts with ODS in several ways:

- using the ODS global statement
- using the SAS Explorer window
- using PROC TEMPLATE
- using PROC OUTPUT

The ODS Statement

The ODS Statement allows you to control several aspects of ODS Output:

Output Generation.

Currently, ODS can produce output in three formats:

1. traditional line oriented LISTINGS
2. HTML, SGML and possibly LaTeX
3. a persistent output hierarchy

LISTING is the moniker for traditional green bar output, and is the default.

ODS LISTING CLOSE;

suspends the generation of the lines sent to the SAS Listing file.

```
ODS HTML file="body.html"
         frame="frame.html"
         Contents="cont.html";
```

enables the generation of HTML formatted tables. Three separate files are mentioned.

The file referenced with frame= contains HTML that configures an HTML browser to show two panes of output. The left-hand pane is a clickable table of contents, containing hot links to the respective tables in the right-hand pane. We currently require Internet Explorer, Netscape or another browser that fully supports the HTML3.2 tag set.

The file referenced with file= receives the meat of the generated HTML3.2 <table> ... </table> tag stream. This HTML shows in the right hand frame of the browser. You are free to "switch" body files without changing the contents file – this would reduce download time in situations where the viewer would not visit all the tables in the output.

The file referenced with contents= contains HTML formed from the hierarchy of output objects produced by the SAS procedures. The text is arranged to reflect the hierarchy, and is comprised of HTML links to the respective tables in the body file.

This organization makes the Table of Contents a very convenient "scrolling device". The following output was produced by adding the appropriate ODS HTML FILE=... FRAME=... CONTENTS=...; statement to the SAS code shown on the first page of this paper.

SAS OUTPUT - Microsoft Internet Explorer

File Edit View Go Favorites Help

Address: D:\Paul\TALK\Pharma 97\glm1.f.html Links

Table of Contents

1. Glim

Data

Class Levels

Number of Observations

Analysis of Variance

x

Overall ANOVA

Fit

Statistics

Type I

Model ANOVA

Type II

Model ANOVA

Type III

Model ANOVA

Type IV

Model ANOVA

y

Overall ANOVA

Fit

Statistics

Type I

Model ANOVA

Type II

Model ANOVA

Type III

Model ANOVA

Type IV

Model ANOVA

The SAS System

General Linear Models Procedure

Dependent Variable: x

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	53	89968.3813	1320.1581	1.41	0.0702
Error	101	94586.5014	936.5000		
Corrected Total	154	184554.8826			

R-Square	C.V.	Root MSE	x Mean
0.425198	3079.919	30.60229	0.993807

Source	DF	Type I SS	Mean Square	F Value	Pr > F
a	3	3477.20727	1159.06909	1.24	0.3001
b	3	2463.81327	821.27109	0.88	0.4557
a*b	9	11782.28799	1309.14311	1.40	0.1990
c	3	9276.62885	3092.20962	3.30	0.0234
a*c	9	21563.48549	2395.94283	2.56	0.0108
b*c	9	9100.23528	1011.13725	1.08	0.3843
a*b*c	17	12304.72309	723.80724	0.77	0.7195

Source	DF	Type II SS	Mean Square	F Value	Pr > F
a	3	6062.23546	2020.74515	2.16	0.0977
b	3	3515.73325	1171.91108	1.25	0.2952
a*b	9	19695.04391	2188.33821	2.34	0.0185

Done

Other Markup Supported

We can generate SGML markup using the SASOUT DTD proposed to SGML Open, and we are in the process of researching how to generate acceptable LaTeX – we'd be interested in your comments on the need for other markup languages.

Output Selection and Exclusion

There are times when one does not want all the information printed by a SAS procedure. To continue to use our GLM example here, suppose we are only interested in the Overall Anova Table, and would like to suppress all other printed output. We can add ODS TRACE OUTPUT; to our SAS job and see the names of the output as they get added, or we can consult the explorer to discover that this table is called "overall".

ODS LISTING SELECT OVERALL; causes the output delivery system to suppress all output whose name is not "overall" while generating LISTING output; the EXCLUDE form of the syntax is more convenient when you are keeping more than you are dropping. SELECT and EXCLUDE can be specified for any of the output destinations supported by ODS.

Persisted [Saved] Output

Another new form of saving your output is the Output Hierarchy itself – actually saving the raw output objects produced by SAS procedures into a new file type, and being able to reuse those objects later.

The SAS Explorer window allows you to navigate this hierarchy and do interesting things to the items that make up the leaf nodes of the tree [typically output objects]. One can select a table and from a popup menu and choose to:

- regenerate the HTML or listing output
- convert the data represented in the table into a SAS data set
- move or copy the table to another location in the hierarchy,
- edit the properties of the table.

PROC OUTPUT allows you to navigate the hierarchy via SAS statements. The "line mode" features of PROC OUTPUT allow the programmer to make similar requests in a canned program requiring no operator intervention.

This form of output is currently useful for folks who would like to generate a set of output and then inspect the results in that output to determine whether or not to print/process it further.

The Output Hierarchy is a central component of future plans for ODS. For example, it can be produced on one host computer (MVS, UNIX) and sent to a PC for interactive perusal, or incorporated into word processing applications.

Output as SAS Datasets

One of the goals of ODS is to obsolete PROC PRINTTO as a means for capturing statistics produced by procedures into data sets. ODS OUTPUT *output_name* = *libname.memname*; will convert the data behind a piece of output into a SAS dataset. The columns will have suitable names, formats and labels.

Customizing Your Output

So far we have shown you how you can have your output in several formats, and how you can select just a part of that output while discarding others. How does the SAS programmer *change* the output produced by procedures? ODS addresses this on two levels. The substance of tabular reports like headings, column headings and widths, formats and justification are kept in Table templates. Stylistic information like color, point size, font names and so on are kept in Palette templates. PROC TEMPLATE is used to manipulate all flavors of templates.

Table Templates

Table Templates are the "report source code" of the output generated by SAS procedures. These templates [like other objects created and maintained with PROC TEMPLATE] participate in an inheritance scheme. For example, many procedures print out values as a matrix, with row and column names. These procedures inherit the definition of the Common.Matrix table template when defining their own templates.

The syntax of a template that describes a table is shown in the first example where we walk through the steps reducing the verbosity of a table printed by PROC GLM.

Palettes and Styles

Palettes and Styles are used to bind abstract stylistic elements to concrete values while generating output. Currently most of the elements in a palette are HTML specific – this will change as we build support for printing ODS output directly to other output devices. Here is a section of the default palette currently defined:

```

proc template;

  define font
    Fonts.Default.DocFont;
    face=Arial;
    size=3;
    weight=medium;
    style=normal;
  end;

  define font
    Fonts.Default.HeadingFont;
    parent = Fonts.Default.DocFont;
    weight=bold;
  end;

  define font
    Fonts.Default.StrongFont;
    parent = Fonts.Default.DocFont;
    weight=bold;
    size=5;
  end;

  ... more bindings ...

  define palette Palettes.Default;

    DocFont = Fonts.Default.DocFont;
    ContentsImage = "altgrey02.jpg";
    PagesImage = "altgrey02.jpg";
    DocImage = " ";
    DocBackGround = "#E0E0E0";
    DocForeGround = "#002288";
    DocVlink = "#004488";
    DocLink = "#0066AA";

    ... more bindings ...

  end;

```

There several interesting things to note about the objects created and managed by PROC TEMPLATE. One of the most useful features is the parent= syntax that allows one object to inherit from another. This allows users to build on the palettes supplied with SAS software instead of having to redefine them from scratch. For example, to override the background and foreground colors for all HTML generated by a SAS session, the programmer would code:

```

Proc template;

  Define palette my_scheme;
    Parent = Palettes.Default;
    DocBackGround = "black";
    DocForeGround = "white";
  End;

  Run;

ODS HTML ... palette = my_scheme;

... regular SAS Statements ...

```

Style templates allow the user to create and reference an entirely new set of palette tags. Style templates can be referenced from a table template to describe the look and feel of any data cell. For example, to create a custom style that can be applied to an out of range data cell:

```

Proc template;

  Define style my_style;
    Parent = Styles.DataCell;
    ForeGround = OutOfRangeForeColor;
    BackGround = OutOfRangeBackColor;
    Font = OutOfRangeFont;
  End;

  Define palette my_scheme;
    Parent = Palettes.Default;
    OutOfRangeForeColor = "Red";
    OutOfRangeBackColor = "Grey";
    OutOfRangeFont =
      Fonts.Default.StrongFont
  End;

  Define table my_table;
    Define column a;
      /* The value 9999 is treated */
      /* as an out of range value */
      Style 9999 = my_style;
    End;
  End;

  Run;

```

Templates, Palettes, and Styles can be stored in permanent SAS data libraries (including SASHELP if you have the appropriate permissions). This makes it easy for all programmers working on a given project to share the same templates, giving generated output a consistent look and feel that can be easily changed just by tweaking the central copy.

Example: Redefining a Table

We would like to make the "overall" table from PROC GLM less verbose – reduce the number of decimals printed, and use terse column headings.

```
/* print only the overall table */
ods listing select overall;

options ls=72 ps=40;

/* show the default table */
ods path sashelp.tmplmst(read);
proc glm data=a;
  class a b c;
  model y = a|b|c / ss1 ss2 ss3 ss4;
run;
```

Dependent Variable: y

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	56	492.5623493	8.7957562	10.47	<.0001
Error	105	88.1757008	0.8397686		
Corrected Total	161	580.7380501			

Now lets redefine the template for this overall anova table...

```
/* use a temporary file for templates created */
ods path work.mytemp(update) sashelp.tmplmst;

proc template;
  define table Stat.GLM.OverallANOVA;

    column source df (ss ms) (fvalue probf);
    underline col_space_min=1 col_space_max=1;

    define source; parent=Stat.GLM.ANOVA.source; width=16; end;
    define df ; parent=Stat.GLM.ANOVA.df; format=4.0; end;
    define ss ; parent=Stat.GLM.ANOVA.ss; header="SS/MS" format=7.2; end;
    define ms ; parent=Stat.GLM.ANOVA.ms; format=7.2; end;
    define fvalue; parent=Stat.GLM.ANOVA.fvalue; header="F/p>F" format=7.2; end;
    define probf ; parent=Stat.GLM.ANOVA.probf; format=pvalue5.2; end;
  end;
run;

/* show the table again */
proc glm data=a;
  class a b c;
  model y = a|b|c
    / ss1 ss2 ss3 ss4;
run;
```

Dependent Variable: y

Source	DF	SS/MS	F/p>F
Model	56	492.56	10.47
Error	105	88.18	<.01
Corrected Total	161	580.74	

Example: Combining Tables

We would like to display a single table showing the "PR > F" column captured from the four tables produced by the GLM / ss1 ss2 ss3 ss4 options.

Our strategy will be to capture the Model Tables printed by PROC GLM into SAS data sets, merge those data sets together and then print the result.

```
data a;
  do a = 1 to 4;
    do b = 1 to 4;
      do c = 1 to 4;
        n = int(6*ranuni(1));
        do i = 1 to n;
          y = a + b + rannor(1);
          output;
        end;
      end;
    end;
  end;
run;

/* no "traditional" output */
ods listing close;

/* but i'd like these output objects */
/* as SAS data sets, please... */
ods output modeltype1=work.model1
           modeltype2=work.model2
           modeltype3=work.model3
           modeltype4=work.model4
           ;

/* go run the model */
proc glm data=a;
  class a b c;
  model y = a|b|c
        / ss1 ss2 ss3 ss4;
run;
```

Now we can merge the 4 data sets together, and create a new variable the reflects the type of model the P-value represents.

```
data work.model;
  set
    work.model1(in=m1 keep=source probf)
    work.model2(in=m2 keep=source probf)
    work.model3(in=m3 keep=source probf)
    work.model4(in=m4 keep=source probf)
  ;

  if m1 then type=1;
  else if m2 then type=2;
  else if m3 then type=3;
  else if m4 then type=4;
run;
```

With all the data in one data set, we'll use PROC TABULATE to generate a summary table.

```
ods html file='g2.html';

title 'The Probabilities are Endless';

proc tabulate data=work.model
              order=data;

  class source type;
  var   probf;

  table source,
         probf="Pr > F for Model Type"
         *type=' '
         *sum=' '
         *f=pvalue8.4
         ;

run;

ods html close;
```

file:///D:/Paul/TALK/Pharma 97/g2.html - Netscape

File Edit View Go Communicator Help

Bookmarks Location: file:///D:/Paul/TALK/Pharma 9

The Probabilities are Endless

		Pr > F for Model Type			
		1	2	3	4
SOURCE	a	<.0001	<.0001	<.0001	<.0001
	b	<.0001	<.0001	<.0001	<.0001
	a*b	0.0295	0.0153	0.0075	0.0872
	c	0.4595	0.4595	0.6489	0.2343
	a*c	0.3410	0.1337	0.2346	0.3959
	b*c	0.0721	0.0721	0.0694	0.3702
	a*b*c	0.4259	0.4259	0.4259	0.4259

Document Done

Example: Augmenting Tables

We would like to add a column to the "model" table produced by GLM. In the words of your Boss:

"computed as Number-of-Rows * ProbF. Thus, since there are 7 rows in this case, the multiplicity adjusted p-value for a*b is $7 * 0.0295 = 0.2065$. This is sort of an *honest* p-value, given that you're scanning all 7 of 'em for significance."

[disclaimer – we are not real statisticians and don't really know if this is a valid transformation to make on these probabilities – this is only an example of what one can do to keep a Boss happy...]

The First step is to capture the model table into a data set and to add the "multiplicity adjusted" column our boss desires.

```
/* no printout for now */
ods listing close;

/* but please create a SAS dataset */
ods output modeltype1=work.model;

proc glm data=a;
  class a b c;
  model y = a|b|c
        / ssl;
run;

/* use PROC SQL to add the new
/* column representing the adjusted
/* probability */
proc sql;

  create table work.model2 as
  select *, probf * count(*) as m_probf
  from work.model;
```

That wasn't too hard. Now spin the report out as HTML. We'll use smaller column widths and terse column headings to create a more punchy report.

```
/* request html generation */
ods html file="report.html";

/* generate the report */
proc report nowd data=work.model2
  headskip headline;

column source df ss ms
       fvalue probf m_probf;

define source / display
  "Source" width=10;

define df / display
  format=4.0;

define ss / display
  "Type I" "Sum of" "Squares"
  format=7.2;

define ms / display
  "Mean" "Square"
  format=7.2;

define fvalue / display
  "F"
  format=6.2;

define probf / display
  "Pr > F"
  format=pvalue6.4;

define m_probf / display
  "Modified" "Pr > F"
  format=pvalue6.4 width=9;

run;
```

This report is displayed in a browser as shown below:

Source	DF	Type I Sum of Squares	Mean Square	F	Pr > F	Modified Pr > F
a	3	222.98	74.33	88.51	<.0001	<.0001
b	3	211.10	70.37	83.79	<.0001	<.0001
a*b	9	16.44	1.83	2.17	0.0295	0.2063
c	3	2.19	0.73	0.87	0.4595	3.2168
a*c	9	8.82	0.96	1.14	0.3410	2.3888
b*c	9	13.78	1.53	1.82	0.0721	0.5050
a*b*c	20	17.44	0.87	1.04	0.4259	2.9811

Work In Progress

We are working at a frenzied pace to deliver Version 7 to our customers during 1998 – there is still time for us to make a few changes so don't hold us to the exact syntax depicted in these examples.

The Usual Caveats

Limited Conversion

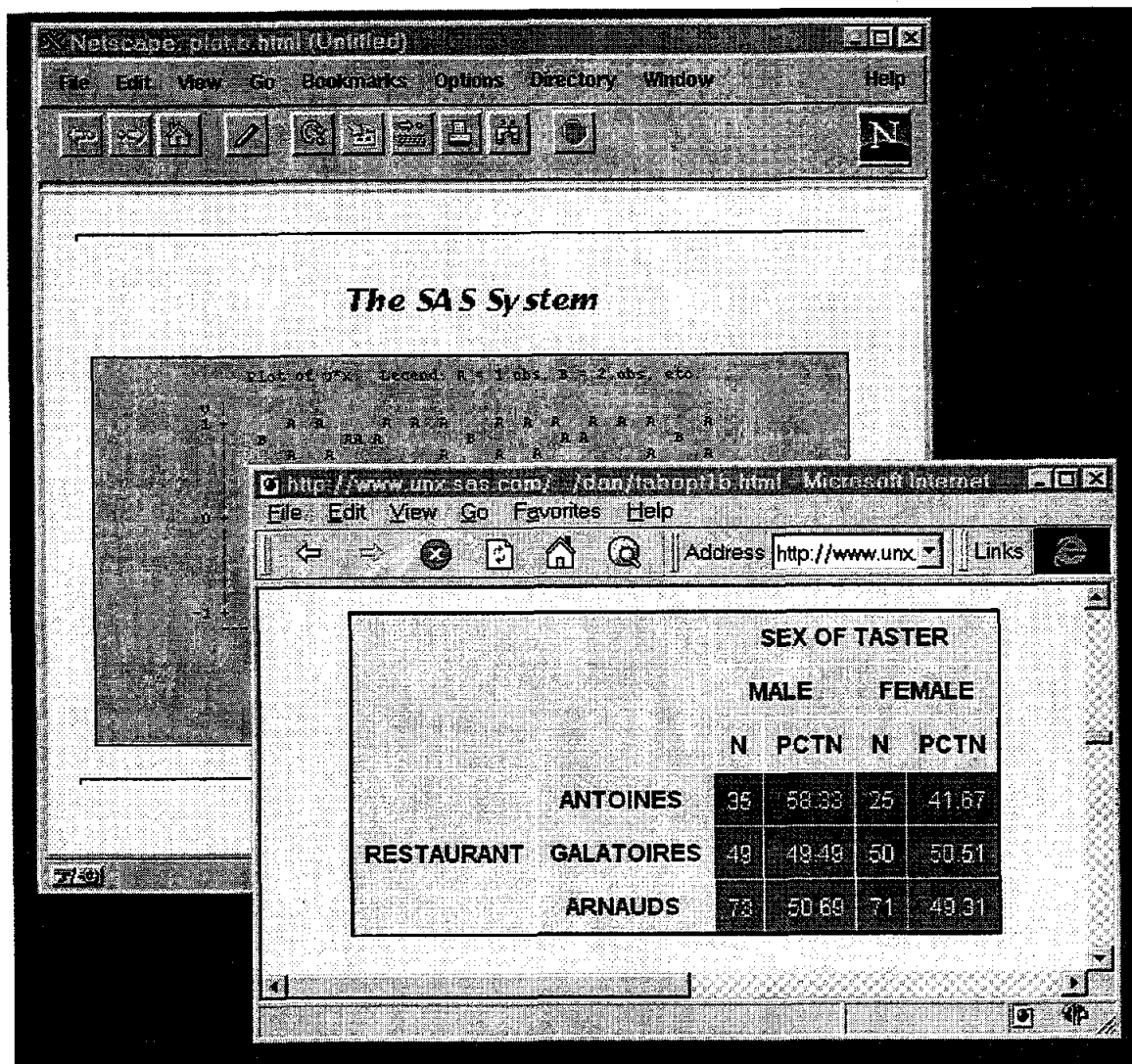
Some SAS procedures were designed to “draw pictures” on a line printer. These procedures have been converted to use ODS, but the generated HTML for example is a verbatim copy of the line mode output.

PROC PLOT, PROC CHART and PROC TIMEPLOT generate “line printer style” HTML. If your site licenses SAS/GRAPH® software, then we

suggest you use PROC GCHART and PROC GPLOT together with the GIF device drivers. HTML generated by ODS will include a reference to this SAS/GRAPH output as an `<IMG SRC=...>` tag, so your graphs will display interspersed with the rest of your HTML output.

PROC CALENDAR and PROC COMPARE will probably get this “line printer” treatment in the first release of Version 7 too – we will upgrade the HTML capability of these PROCs based on user demand

All other procedures including PROC FREQ, PROC REPORT and PROC TABULATE are fully converted, and generate high fidelity HTML. The screen dump inserted here shows the “line printer” treatment used for PROC PLOT in the background as well as the high fidelity treatment for PROC TABULATE.



Compatibility Issues with Version 6 ODS Prototype

In Version 6 prototype PROC MIXED uses a MAKE statement to create datasets from output objects. This statement remains supported in MIXED, but all other procedures achieve this objective with the global ODS OUTPUT statement.

In the Version 6 prototype the output hierarchy was stored in a SAS catalog. Version 7 of SAS Software has a more flexible *item-store* file type that is used to store templates, palettes and a saved output hierarchy. There are currently no plans for a conversion procedure.

In the Version 6 prototype the saving of an output hierarchy was controlled by the `_DISK_` and `_PRINT_` macro variables. Version 7 uses the global ODS Statement to indicate this.

The Version 6 and Version 7 PROC TEMPLATE and PROC OUTPUT syntax are not compatible.

Future Work Items

Hardcopy High Fidelity Printing

We realize that printing HTML as viewed by a browser leaves a lot to be desired when the reports are longer or wider than a physical page. We are beginning to research and develop the code necessary to print SAS output as Postscript and PDF documents directly.

Output directly into Word Processing Application

We would like to implement drivers that would deposit ODS output directly into a Microsoft Word document. Are there other important formats you would like us to consider here?

Other Access to Output Hierarchy

SAS/AF® Software applications developers would like to query the contents of an output hierarchy and retrieve the data behind the output objects stored in the hierarchy from their SCL programs. They'd like to display the data in the hierarchy in the SAS/AF table widgets.

Programs external to SAS would like to query the output hierarchy via something like a Microsoft DCOM binding, or a Java interface.

Your Turn

The developers, testers and documentation folk that bring you ODS are very excited about the potential of these new capabilities of the SAS System. You can send us electronic mail to ods@unx.sas.com with your comments, or to request a copy of the example HTML files generated by a broad selection of SAS procedures, and in a variety of styles.

We hope to add more information about the ODS capabilities in future releases of SAS Software to the Research and Development section of SAS Institute's Web Page. See <http://www.sas.com/rnd>

Trademarks

SAS, SAS/GRAPH and SAS/AF 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.