

Feeding Data _NULL_ to the Lions: Bringing ODS MARKUP under the Big Top with Adobe Acrobat 5 Forms

Thomas E. Kunselman, California State University, Stanislaus

Abstract

Creating customized reports using Data _NULL_ has always been a tedious process of counting columns and adjusting put statements. With the addition of ODS MARKUP in SAS® Version 8.2, it is now possible to use a WYSIWIG editor such as MS Word or Adobe PageMaker to lay out your SAS reports and use PROC PRINT to generate them. This is done by converting the WYSIWIG document to an Acrobat Form and filling in the form with a PROC PRINT and ODS MARKUP using a customized tagset to create the Adobe Form Data File (FDF). Among the advantages of using an Acrobat Form are eliminating clerical layout tasks for programmers, no truncated data: the values in a text field can automatically shrink to fit in the field size, forms data can be modified and updates submitted back to a Web server, and best of all professional looking output consistent with other ODS destinations. This paper is intended as an exploration of the synergistic possibilities between SAS and Adobe Acrobat Forms.

Introduction

Adobe Portable Document Format (PDF) provides a means for exchanging professional looking documentation across many platforms using the Adobe Acrobat software. The Adobe file format preserves fonts, formatting and graphics of the original source document which can be viewed and printed in a way that looks exactly the same no matter what the platform. Using the Output Delivery System (ODS) in SAS Version 8.2 to create PDF output has virtually eliminated the need for additional formatting of procedure output before distribution on paper or via e-mail.

PDF files also make a nice replacement for HTML files, eliminating the worry about how your manually created and ODS generated HTML might look in one browser vs. another. But now that SAS PROC output can be created in a PDF format what about those old DATA _NULL_ reports that use a lot of pointer control to move things around? After learning of ODS Markup in Version 8.2 one solution that suggested itself was the creation of a tagset to generate Adobe Acrobat Form Data Files (FDF) from SAS and use this to fill in Acrobat Forms. And the next question that arises from displaying SAS data in Acrobat forms is, "What about using Acrobat to input information too?" This paper will explore some of the possibilities for linking SAS and Adobe Acrobat Forms.

Adobe Acrobat Version 5 Functionality

Following is a list of Adobe Acrobat features and some advantages that make it a worthwhile interface tool to use with SAS.

- **Easy to use tools to create professional layout.** The report layout for an observation from a data set can be easily completed by anyone familiar with MS Word or PageMaker. No more counting columns and controlling the pointer with a

DATA _NULL_ step!

- **Form objects.** Any form object available in HTML is available in Acrobat Forms. After the layout document is completed in MS Word or your favorite word processor, conversion of the document to PDF allows form fields to be added in Adobe Acrobat. Please note that this is the full version of Adobe Acrobat, not Acrobat Reader.

- **No more truncated displays of data.** Objects in Acrobat forms can change the font size used to display data automatically. This allows the data to fit completely inside the allotted field space on the form without being truncated.

- **Electronic submission of data.** Acrobat forms allow modifications to be submitted electronically to the web server. Submission must take place within a Web browser.

- **Dynamic Forms.** Javascript is available in Adobe Acrobat 5 and can be used to validate forms, calculate values, insert additional form pages into a document from templates, etc.

- **Reduced programming resources.** The clerical tasks associated with programming the layout of a report could be transferred to general support staff. The appearance of the form can be changed without changing the program allowing custom designs for different business units.

- **Forms Data Format (FDF).** The data in a PDF form can be exported from the form into two formats. FDF is a text file format associated with a PDF. The FDF files contain only the form field data from the PDF form so they are significantly smaller than the PDF. FDF files can also reference PDF templates. If there are multiple observations in the FDF then each observation will insert a new form page from the template into the associated PDF document.

- **XML (XFDF).** Adobe Acrobat has the ability to import or export XML format data defined as XFDF.

Adobe Forms Data Format

FDF is based on PDF and uses the same syntax. The MIME type used by FDF is: type application/vnd.fdf. The three required elements for an FDF are the header, body, and trailer. The header is one line identifying the version number of the FDF specification. Although the current version of PDF is 1.4, the FDF header version should remain 1.2 because Acrobat 3.0 and earlier versions will not recognize an FDF file with a version other than 1.2.

The SAS code in Example 1 creates an FDF file from a data set with a single observation using the XML engine. As part of the FDF file it is possible to specify an associated PDF file. The FDF tagset uses a macro variable, PDFNAME, to contain the name and location of the associated PDF file. As in this example, it is possible to associate the FDF with a URL pointing to the PDF form. If the FDF file is placed on a webserver or streamed to a browser, the URL will be used to load the PDF file as shown in Figure 1.

Example 1

```
%let pdfname =
http://localhost/facultyworkload.pdf;

libname testxml xml 'test.libname.fdf'
tagset=tagsets.fdf;

proc copy in=work out=testxml;
select multiple;
```

```
<</V (ECON 2510)/T (CRN1)>>
<</V (03)/T (S1)>>
<</V (F)/T (LearnMode1)>>
<</V (LEC)/T (Type1)>>
<</V (1)/T (Units1)>>
<</V (3)/T (UnitTotal1)>>
<</V (3)/T (Team1)>>
<</V (3)/T (WTU1)>>
... Variables 2 through 6 ...
<</V (PADM 5006)/T (CRN7)>>
<</V (01)/T (S7)>>
```

Figure 1

A. INSTRUCTION			SL(Y) LearnMode: (S or (A))	Lec/ Lab/ Other	Teaching Units	K Factor	Total Teaching Units	Team Teaching Fraction	Actual WTU's Famed
Course No.	Sec	Course Title							
ECON 2510	00		F	LEC	1		0	0	0
ECON 3100	01		F	LEC	1		0	0	0
ECON 3050	01		F	SEM	1		0	2.1	0
ECON 3050	02		S	SEM	1		0	0.2	0
ECON 3050	04		S	SEM	1		0	0.5	0
ECON 3050	05		S	SEM	1		0	0.2	0
PADM 5006	01		F	SEM	1		0	0	0

run;

The form displayed in Figure 1 was created in MS Word for use as a paper form. This was then converted to a PDF using Adobe Acrobat and the text field boxes were added. For this example the text field boxes are outlined, but their borders could be invisible. Layout of the form consisted of drawing boxes with the cursor and naming them using the Adobe Acrobat 5 form tool.

From the browser the TEST.LIBNAME.FDF file was opened and the pointer to the URL with the form automatically retrieved the FACULTYWORKLOAD.PDF document from the webserver. Example 2 shows the FDF file created using the FDF tagset and SAS XML engine in Example 1. Each of the fields is in the form of <</V (value)/T (varname)>>. For brevity, the fields from 2 through 6 are omitted in the FDF source. In the FDF body the /F tag can contain a value that can be a filename or, in this case, a URL.

Example 2

```
%FDF-1.2
1 0 obj
<<
/FDF <<
/Fields
[
<</V (235)/T (dept)>>
<</V (27)/T (College)>>
<</V (EJPeterson)/T (facultyname)>>
<</V (2004)/T (TERM)>>
<</V (0001)/T (workload)>>
```

```
<</V (F)/T (LearnMode7)>>
<</V (SEM)/T (Type7)>>
<</V (1)/T (Units7)>>
<</V (3)/T (UnitTotal7)>>
<</V (3)/T (Team7)>>
<</V (3)/T (WTU7)>>
]
/F (http://localhost/
facultyworkload.pdf)
>>
>>
endobj
trailer
<< /Root 1 0 R >>
%%EOF
```

Acrobat Forms 5 can now use an XML version of the FDF called XFDF to fill in a form. There are two disadvantages of using XFDF. Templates cannot be accessed with XFDF so it is impossible to send multiple observations for the same form. If multiple observations are written to an FDF referencing a template, the Acrobat Form will dynamically generate as many form pages as required for the number of observations. If an XFDF file contains multiple observations the template cannot be accessed and the form objects will take the value of the last observation. Another disadvantage of XFDF is the inability to save digital signature information in the XFDF file.

Electronic Submission of PDF Forms Data to SAS

The previous example demonstrated how to extract data from SAS and push it into an Acrobat form. In the next example, the Acrobat form will electronically submit data to the SAS/IntrNet® application dispatcher in the same way an HTML form would submit data. Figure 2 shows an Acrobat form converted to PDF from an MS Word document. The survey in MS Word was originally used to print a paper survey that was then mailed to students via the US Postal Service. It took less than a day to convert this paper survey to a web survey using Adobe Acrobat forms and SAS/IntrNet®.

Figure 2

The screenshot shows a web browser window with the address bar displaying 'http://localhost/P169 - Alumni ...'. The browser's menu bar includes 'File', 'Edit', 'View', 'Favorites', 'Tools', and 'Help'. The address bar contains 'http://localhost/P169 - Alumni ...'. The browser's toolbar includes 'Back', 'Forward', 'Home', 'Search', 'Go', 'Links', and 'Norton AntiVirus'. The browser's status bar shows '100%' zoom. The main content area displays a survey form titled 'CALIFORNIA STATE U Alumni'. The form includes a title, a paragraph of text, and a section titled 'PART A. QUALITY OF INSTRUCTION'. Below this, there is a list of eight questions (a-h) with radio button options for 'At the University?' and 'In my major?'. The 'In my major?' column has several radio buttons selected. A 'Submit Survey' button is at the bottom.

When the "SUBMIT SURVEY" button is pressed Acrobat passes all of the variables to the SAS/IntrNet application dispatcher. Example 3 shows the results of the variables passed to the application dispatcher using the `_debug=131` parameter. Once the survey data has been passed to the application dispatcher the values from the macro variables are written to a permanent SAS data set.

Example 3

Symbols passed to SAS

```
#symbols: 37
  "_SRVNAME" = "localhost"
  "_SRVPORT" = "80"
  "_REQMETH" = "POST"
  "_RMTHOST" = "127.0.0.1"
  "_RMTADDR" = "127.0.0.1"
  "_RMTUSER" = ""
  "_HTCOOK" = ""
  "_HTUA" = "Mozilla/4.0 (compatible;
Adobe Acrobat Control Version 5.00 for
ActiveX)"
  "_mrvimg" = "/sasweb/IntrNet8/MRV/
images"
  "_grfaplt" = "/sasweb/graph/
graphapp.jar"
  "_grafloc" = "/sasweb/graph"
  "_debug" = "131"
  "_program" = "webpgm.submitpdf.sas"
  "_service" = "default"
  "Q1AM" = "3"
  "Q1AU" = "4"
  "Q1BM" = "4"
  "Q1BU" = "3"
  "Q1CM" = "2"
  "Q1CU" = "1"
  "Q1DM" = "3"
  "Q1DU" = "2"
  "Q1EM" = "3"
  "Q1EU" = "2"
  "Q1FM" = "2"
  "Q1FU" = "3"
  "Q1GM" = "3"
  "Q1GU" = "1"
  "Q1HM" = "4"
  "Q1HU" = "4"
  "submit" = ""
  "_VERSION" = "8.2"
  "_URL" = "/cgi-bin/broker.exe"
  "_ADMIN" = "Thomas Kunselman"
  "_ADMAIL" =
"Tkunselman@stan.csustan.edu"
  "_SERVER" = "localhost"
  "_PORT" = "5001"
```

A limitation of using this method is that the total URL length is limited to 2083 characters on Windows and 1023 characters on the Mac. Acrobat defaults to excluding empty form fields on the URL and allows selected fields to be defined for inclusion or exclusion when posting data. Acrobat forms are also able to submit forms data via FDF or XFDF files. These files could then be parsed on the server side. Adobe supplies a Forms Data Format Toolkit for parsing FDF data files streamed from a browser to a web server.

Conclusion

My interest in Adobe Acrobat started with the availability of the ODS PRINTER destination in Version 8 software. In fact, the experimental PDF release in 8.1 caused me to invest in the full Adobe Acrobat product in order to distill Postscript into PDF's. My main reason for using ODS is to reduce the amount of effort going into manually formatting and then proofing data for publication. This allows more time for analyzing the data rather than checking for typographical errors.

While most of my legacy reports were easy to convert for ODS output in PDF, I was still worrying about how to get my ancient DATA_NULL_ reports that use a lot of pointer controls into a professional looking format consistent with the ODS PDF output. About this same time I attended the Advanced ODS pre-SUGI 26 training and learned about TAGSETS and XML. After some false starts, I came across the functionality of Acrobat Forms and realized that a SUGI presentation was the perfect excuse to see if any of this would work. And it does!

The Next Step

There are several additional areas to explore in looking at how to further integrate Adobe Acrobat with SAS and SAS/IntrNet.

- **Digital Signatures.** After a form is completed by an individual, the FDF file can contain digital signature information which makes it possible to recreate the signed document on the server.

- **Improve FDF Tagset.** Improve the FDF tagset so that it can be used with other PROCs and not just the XML engine. This may have to wait until SAS V9. An improved tagset will also allow streaming of FDF files to the browser via SAS/IntrNet and _webout rather than relying on ftp and URL redirection to the FDF.

- **Multiple Observations.** Improve the FDF tagset so it can be used to dynamically synthesize PDF documents composed of a variable number of pages, from templates found in PDF documents specified by the FDF, and to populate any fields in the "spawned" pages with data.

- **Parse FDF and XFDF input files.** Create SAS programs or perhaps XML maps to import FDF and XFDF data files.

Acknowledgments

I want to thank the SAS Institute ODS developers and technical support people for all of their help in getting to know the Output Delivery System.

I especially want to thank Brian Schellenberger for his much appreciated help in regards to ODS PDF.

References

Adobe Forms Data Format Toolkit Overview and Reference, Technical Note #5194: Version: Acrobat 5.0, Adobe Systems Incorporated, September 2001

Adobe Forms Data Format Toolkit Overview, Technical Note #5191: Version: Acrobat 5.0, Adobe Systems Incorporated, June 2001

Forms System Implementation Notes, Version: Acrobat 5.0, Adobe Systems Incorporated, Revised January 25, 2001

<http://www.sas.com/rnd/base/topics/odstagsets> (Date: 17SEP2001 "Creating Customized Tagsets to Use with ODS and XML LIBNAME ENGINE")

<http://www.sas.com/rnd/base/topics/templateFAQ/xmlf.htm> (Date: 17SEP2001 "XML, ODS and The Markup Language")

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc., Cary, NC, USA.

Contact Information

To request the ODS FDF template used in this paper please send an e-mail request to tkunselman@stan.csustan.edu.

Thomas Kunselman has spent over 15 years using many of the SAS system products. He has experience creating web-based reporting applications with varied types of data, including higher education, health care, pharmaceutical, and manufacturing process data.

Thomas Kunselman
 Manager, Data Warehousing and Analytics
 California State University, Stanislaus
 801 West Monte Vista Avenue
 Turlock, CA 95382
tkunselman@stan.csustan.edu
 Voice: 209.667.3281
 Fax: 209.667.3251
http://www.csustan.edu/Inst_Research