Paper 157-30

Make Bill Gates and Dr. Goodnight Run Your SAS Code: Using VBA, ADO and IOM to Make Excel and SAS Play Nice

Ted Conway, Ted Conway Consulting, Inc., Chicago, IL

ABSTRACT

With the advent of the SAS Integrated Object Model (IOM), which has even found its way into the $125 SAS Learning Edition, SAS has made communication between Excel and SAS more flexible than ever. This paper presents simple examples of how one can write SAS code, run SAS programs, and retrieve the results for presentation in Excel–all without leaving the Excel environment! It may be of interest to anyone who uses Base SAS and Microsoft Excel on the PC platform.

INTRODUCTION

So how do you manage to get data between Excel and SAS?

Flat files?

PROC EXPORT?

PROC IMPORT?

DDE?

OLE?

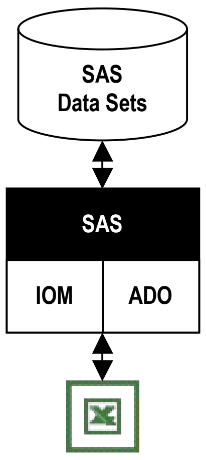
ODBC?

ODS?

All of the above?

Well, with the SAS Integrated Object Model (IOM), Microsoft ActiveX Data Objects (ADO) and Excel Visual Basic for Applications (VBA), you have yet another way to attack this age-old problem!

IOM, ADO AND VBA IN A NUTSHELL



SAS Code Execution

User-written Excel macro code (**VBA**) can create a SAS Integrated Object Model (**IOM)** workspace to establish a connection to a SAS System session that can be used to submit and run SAS code from Excel using the same familiar set of resources and facilities as an interactive or batch SAS session.

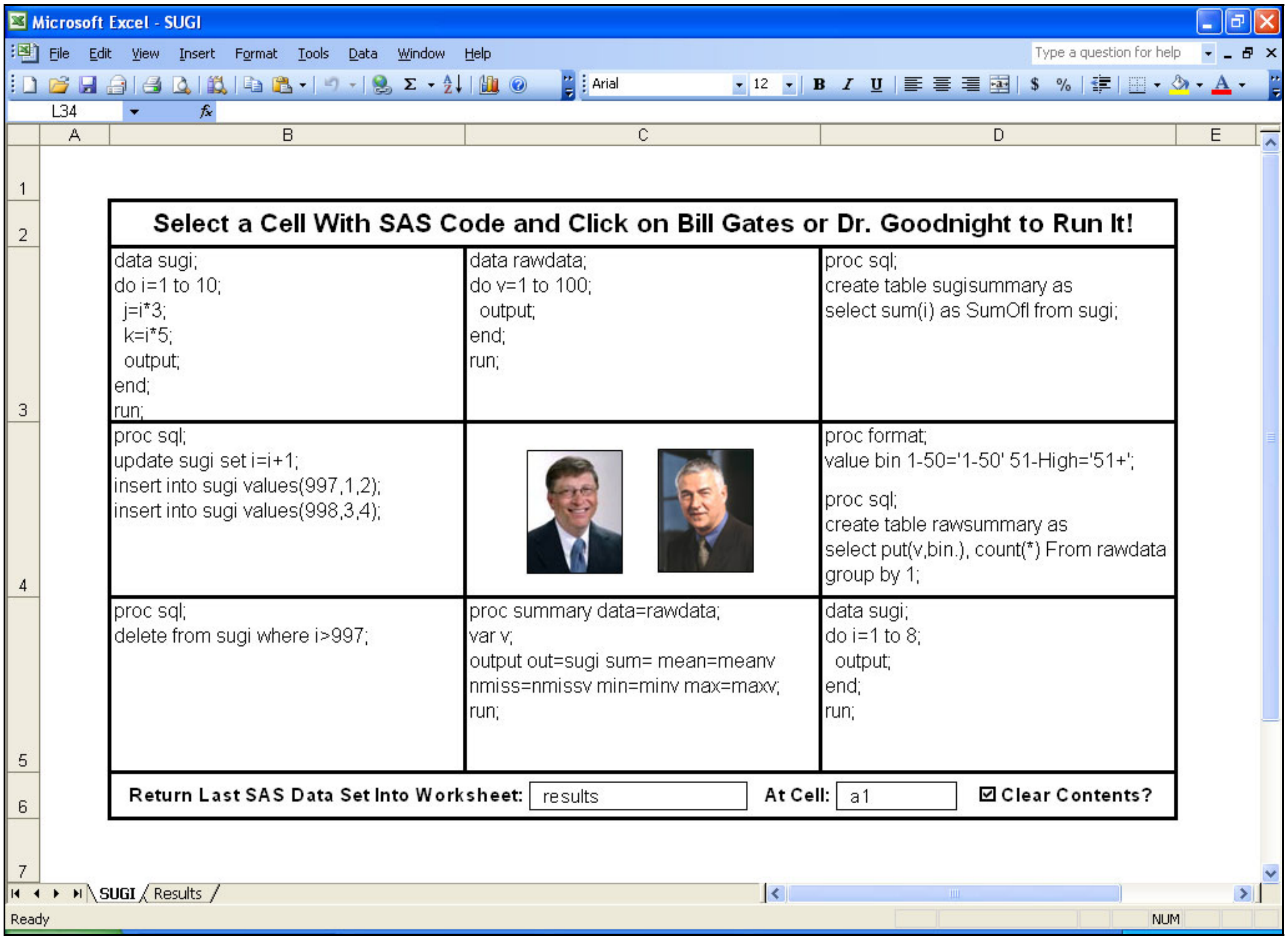
SAS Data Set Access

User-written Excel macro code (**VBA**) can use Microsoft's ActiveX Data Objects (**ADO**) – a language-neutral object model – to access SAS data sets from Excel, returning the data in **recordsets** for subsequent processing.

Excel Formatting and Processing

User-written Excel macro code (**VBA**) can process the **recordsets** returned by **ADO** requests, formatting the data as needed and placing it into the desired locations in existing or new worksheets.

SO HOW ABOUT AN EXAMPLE?



Select a Cell With SAS Code and Click on Bill Gates or Dr. Goodnight to Run It!

```
data sugi;
do i=1 to 10;
  j=i*3;
  k=i*5;
  output;
end;
run;
```

```
data rawdata;
do v=1 to 100;
  output;
end;
run;
```

```
proc sql;
create table sugisummary as
select sum(i) as SumOfI from sugi;
```

```
proc sql;
update sugi set i=i+1;
insert into sugi values(997,1,2);
insert into sugi values(998,3,4);
```

```
proc format;
value bin 1-50='1-50' 51-High='51+';

proc sql;
create table rawsummary as
select put(v,bin.), count(*) From rawdata
group by 1;
```

```
proc sql;
delete from sugi where i>997;
```

```
proc summary data=rawdata;
var v;
output out=sugi sum= mean=meanv
nmiss=nmissv min=minv max=maxv;
run;
```

```
data sugi;
do i=1 to 8;
  output;
end;
run;
```

Return Last SAS Data Set Into Worksheet: results At Cell: a1 ☑ Clear Contents?

There's a wealth of information about SAS, IOM, Excel, ADO, and VBA on the web, but their interaction is best illustrated by a simple working example:

- They say a picture is worth a thousand words, so our example will use two–one of Microsoft Chairman Bill Gates and one of SAS CEO Dr. Jim Goodnight.
- We're going to put the two tech titans to work submitting our SAS code, which will be contained in individual Excel worksheet cells–use as many as you need!
- Click on either exec's PR photo, and the SAS code contained in the currently selected cell will be submitted.
- After the code is run, the contents of the last-created SAS data set (i.e., _last_) will be fetched and placed at the specified location in the requested worksheet.
- Optionally, the contents of the results worksheet can be cleared prior to retrieving the data.
- To create line breaks within a given Excel worksheet cell, use **ALT+ENTER**.
- If you find the selected SAS code obscures your view, you may wish to temporarily hide the Excel Formula Bar.
- In case you're wondering, the Excel format used for Gates' cell is $###,###,###,##0.00 (nyuk, nyuk, nyuk!).

Admit it; it feels pretty good ordering Gates and Goodnight around, doesn't it?

SHOW ME THE CODE!

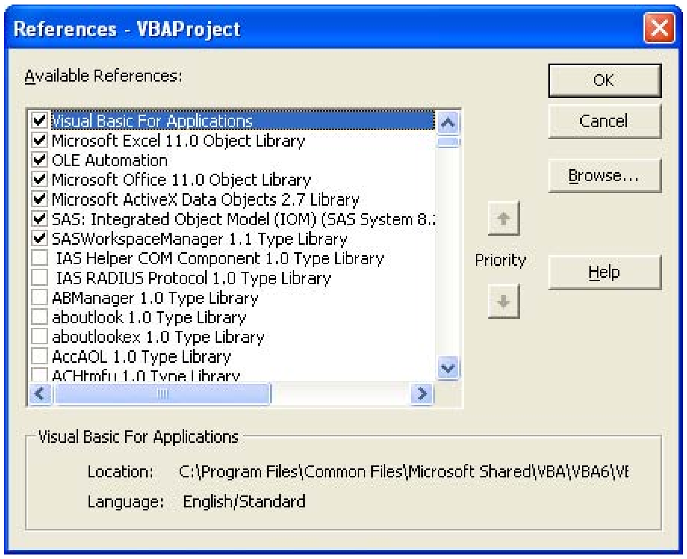
The Excel macro assigned to the images of Bill Gates and Dr. Goodnight, appropriately named **CEO_SUBMIT**, is shown below. **CEO_SUBMIT** is executed whenever either of the two's photos are clicked.

```
'--> SAS IOM Workspace Connection Declarations

Public obWSM As New SASWorkspaceManager.WorkspaceManager
Public obWS As SAS.Workspace
Public obWSflag As Integer

Sub CEO_Submit()

'--> ADO Database Connection Declarations

Dim obConn As New ADODB.Connection
Dim obRS As New ADODB.Recordset
Dim errorstring As String

'--> Create a Local SAS Workspace (One-Time Processing)

If obWSflag = 0 Then
  Set obWS = obWSM.Workspaces.CreateWorkspaceByServer("Local", _
                    VisibilityProcess, Nothing, "", "", errorstring)
  obWSflag = 1
  End If

'--> Submit SAS Code in Active Excel Cell

obWS.LanguageService.Submit ActiveCell.Value

'--> Create an Excel Record Set From The _last_ SAS Data Set

obConn.Open "provider=sas.iomprovider.1; SAS Workspace ID=" + obWS.UniqueIdentifier

obRS.Open "_last_", obConn, adOpenStatic, adLockReadOnly, adCmdTableDirect

'--> Activate and (Optionally) Clear Results Worksheet to Accept Results

Worksheets(Sheets("sugi").ResultsSheet.Text).Activate
If Sheets("sugi").ResultsClear = True Then Cells.Delete

'--> Display Field Names Beginning at Requested Cell

r = Range(Sheets("sugi").ResultsCell.Text).Cells.Row
c = Range(Sheets("sugi").ResultsCell.Text).Cells.Column
For Each fld In obRS.Fields
 Cells(r, c).Value = fld.Name
 c = c + 1
Next

'--> Copy ADO Record Set Below Field Names

r = r + 1
c = Range(Sheets("sugi").ResultsCell.Text).Cells.Column
Cells(r, c).CopyFromRecordset obRS
Cells(r - 1, c).Select

obRS.Close
obConn.Close
End Sub
```

A NOTE ON EXCEL SET UP REQUIREMENTS

To allow Excel to talk to SAS, select Tools►References in the Excel Visual Basic Editor, and make sure that the following three references (or more current versions of the three!) are selected:

- Microsoft ActiveX Data Objects 2.5 Library
- SAS: Integrated Object Model (IOM) 1.0 Type Library
- SASWorkSpaceManager 1.0 Type Library



Available References:

- [✓] Visual Basic For Applications
- [✓] Microsoft Excel 11.0 Object Library
- [✓] OLE Automation
- [✓] Microsoft Office 11.0 Object Library
- [✓] Microsoft ActiveX Data Objects 2.7 Library
- [✓] SAS: Integrated Object Model (IOM) (SAS System 8.:
- [✓] SASWorkspaceManager 1.1 Type Library
- [ ] IAS Helper COM Component 1.0 Type Library
- [ ] IAS RADIUS Protocol 1.0 Type Library
- [ ] ABManager 1.0 Type Library
- [ ] aboutlook 1.0 Type Library
- [ ] aboutlookex 1.0 Type Library
- [ ] AccAOL 1.0 Type Library
- [ ] ACHtmfu 1.0 Type Library

Visual Basic For Applications

Location: C:\Program Files\Common Files\Microsoft Shared\VBA\VBA6\VE

Language: English/Standard

CONCLUSION

By using IOM, ADO and VBA, it's easier than ever to make SAS and Excel "play nice together," which can benefit both you and those you work with.

For the purposes of this paper, things have intentionally been kept relatively simple–use your imagination and you're likely to find many more creative and useful ways to pair SAS and Excel.

For example, you might want to try taming Excel pivot tables with ADO, IOM, and VBA. Or perhaps access remote data sources instead of the local data sources shown in this paper.

You could even use the techniques to in-effect play Excel-SAS "ping-pong", using IOM to allow SAS macros to generate VBA code for Excel, providing real macro processor functionality that Excel still lacks.

ACKNOWLEDGMENTS AND REFERENCES

Be sure to visit the SAS Enterprise Integration Community at http://support.sas.com/rnd/eai/.

The good SAS folks have placed lots of sample code out there that you can "borrow", as I have done above!

Sample code for the ***ADO/IOM Server*** can be found at http://support.sas.com/rnd/eai/samples/adoiom/index.html, while some good examples of code for ***Transferring Data Between SAS® and Microsoft® Excel and Access with Microsoft® Visual Basic® Script*** are available at http://support.sas.com/rnd/eai/samples/VBoffice/index.html.

CONTACT INFORMATION

Ted Conway currently works for Ted Conway Consulting, Inc. (guess how he got that job!) in Chicago, Illinois. He can be reached at tedconway@aol.com.

TRADEMARKS