

Accessing SAS® Data without using SAS Code

by Philip R Holland, Holland Numerics Ltd

Abstract

Recent developments in SAS® for Windows have provided users with routes to SAS data and applications without having to write SAS code using the SAS System. This paper describes two examples of these facilities, ODBC and DDE, which could place SAS at the centre of any application development for the Windows platform.

1. Introduction

In the past SAS has been used to read data from other Windows data sources, eg. Microsoft Access Tables using SAS/ACCESS® for ODBC, and control other external Windows application using DDE (Dynamic Data Exchange). SAS is, of course, a Windows application itself and can now be used as an external application for those other Windows applications. This role reversal expands the range of uses for SAS in the Windows environment in areas where SAS has not been traditionally the first choice application. The ability of SAS to read and maintain data from a wide range of sources can now be utilised throughout the Windows arena.

2. ODBC

The SAS ODBC Driver has been supplied with Base SAS for Windows since Release 6.10 to provide an interface to SAS Data Libraries for other Windows applications. This section will describe the use of the SAS ODBC Driver in Releases 6.11 and 6.12 with Microsoft Access 2.0, Excel 5.0 and Visual Basic 4.0. Each application has its own particular uses and limitations for the ODBC interface.

Single ODBC access of SAS data on the same machine as the user uses the PROC ODBCSEV procedure, which is supplied with Base SAS, running in a single SAS region. Multiple ODBC access of SAS data, or ODBC access on a remote machine, requires SAS/SHARE®, and possibly SAS/SHARE*NET™ as well.

2.1 Setting up a SAS Server for the ODBC Driver

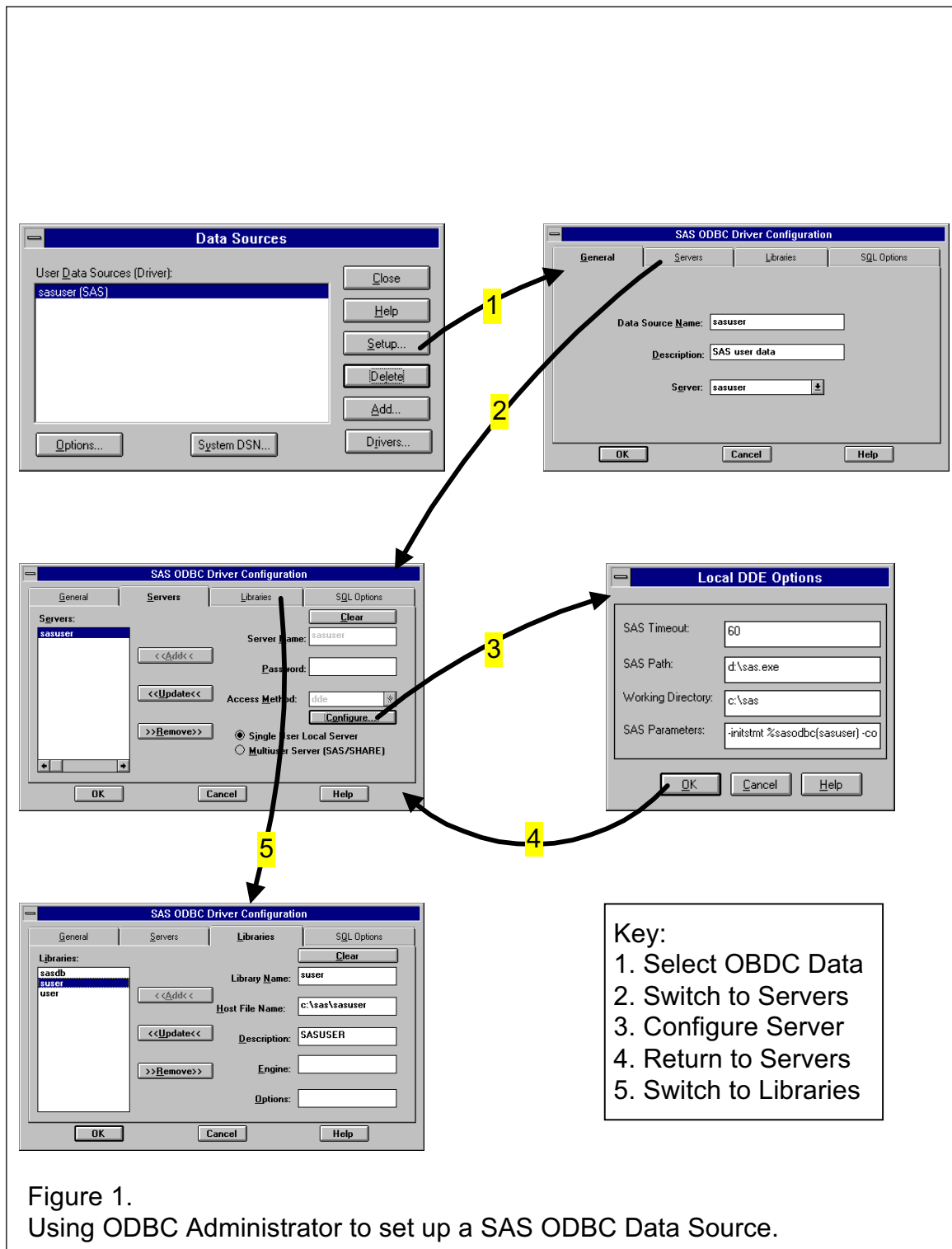
It is very important to plan, in advance, which SAS Data Libraries will be accessed via the SAS ODBC Driver, as the LIBNAME statements must be defined using the ODBC Administrator application found in the Control Panel. In particular, for any ODBC Data Source, there can only be one libname that can be written to by an external application, ie. USER, as Microsoft Access, and similar applications can only write to tables with a single level table name. This name, say XYZZY, would be assumed to be the table WORK.XYZZY, except that the USER library name will override the WORK library name, allowing permanent SAS Datasets to be created.

Other features of the ODBC Data Source definitions include:

- Server names (note that under Win32s, there can only be a single SAS session active at any time, but Windows 95 and Windows NT permit multiple SAS sessions to be active concurrently).
- Command line options when invoking SAS (eg. -AUTOEXEC, -NOLOGO, etc.).
- SAS Data Library names used for importing into external Windows applications.
- Changes to the libnames in a running ODBC Server can be made for subsequent ODBC connections.

It should be noted that the SAS ODBC Driver is unable to read the supplied SAS libnames, MAPS, SASUSER and SASHELP, using their own special libnames. If these libraries need to be read, they should be allocated to different libnames in ODBC Server Libraries panel,

eg. SMAPS, SUSER and SHELP.



The Setup procedure is as follows (see Figure 1):

- Select User Data Sources (Driver), eg. 'sasuser (SAS)'.
- Click Setup.
- Select Data Source Name, eg. 'sasuser'.
- Select Description, eg. 'SAS user data'.
- Select Server, eg. 'sasuser'.

- Click Servers tab.
- Select Server Name, ie. 'sasuser'.
- Click Single User Local Server.
- Click Configure.
- Select SAS Path, eg. 'd:\sas.exe'.
- Select Working Directory, eg. 'c:\sas'.
- Select SAS Parameters, ie. '-initstmt %sasodbc(sasuser) -comamid dde -icon'.
- Click OK.
- Click Libraries tab.
- Select Library Name, eg. 'suser'.
- Select Host File Name, eg. 'c:\sas\sasuser'.
- Select Description, eg. 'SASUSER'.
- Click <<Update<<.
- Repeat selections as required.
- Click OK.
- SAS ODBC Driver is now setup for use.

2.2 Microsoft Access 2.0

Microsoft Access can use the SAS ODBC Driver to read from SAS Data Libraries with the Import menu option, or write to a specific SAS Data Library with the Export menu option.

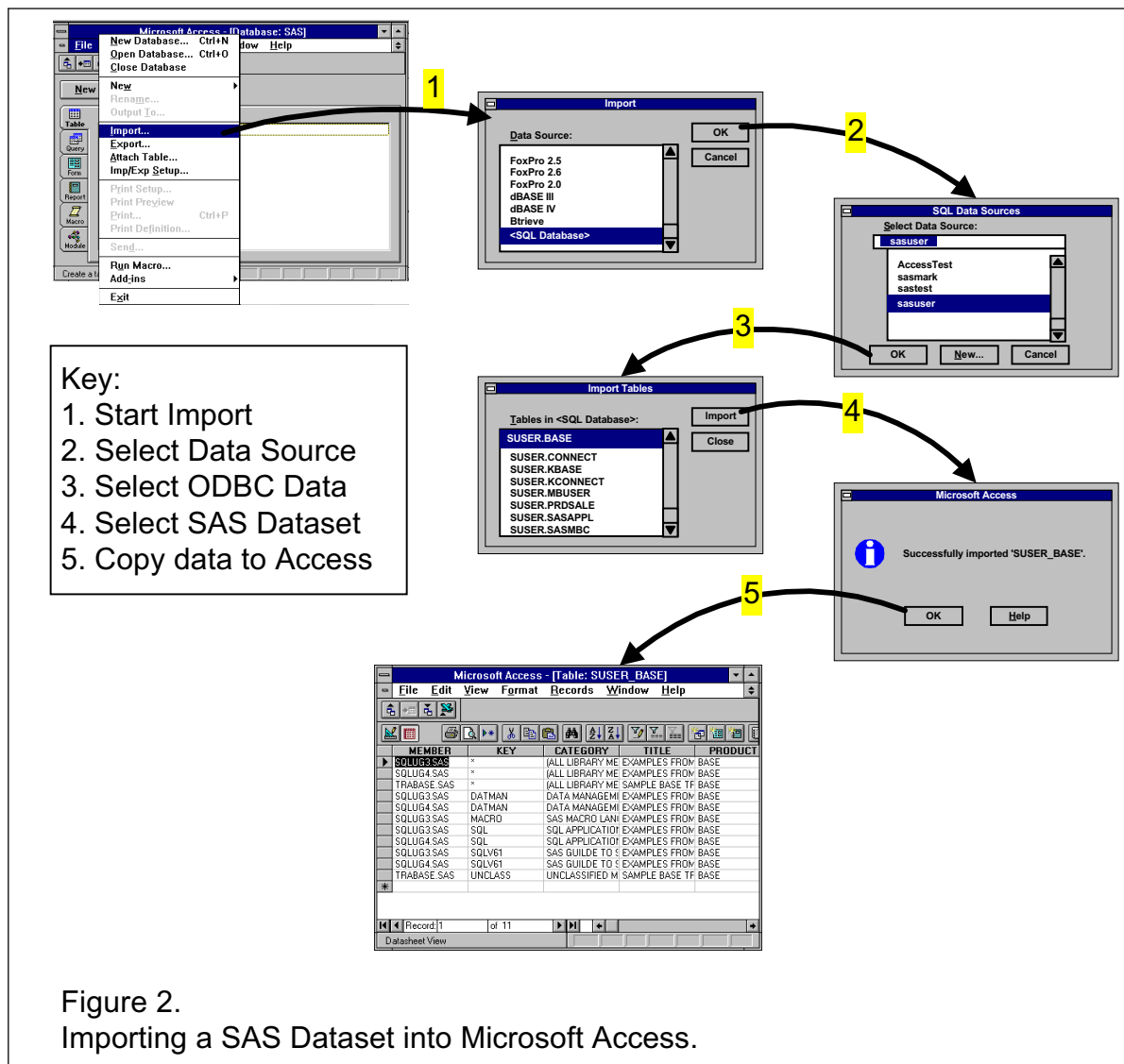
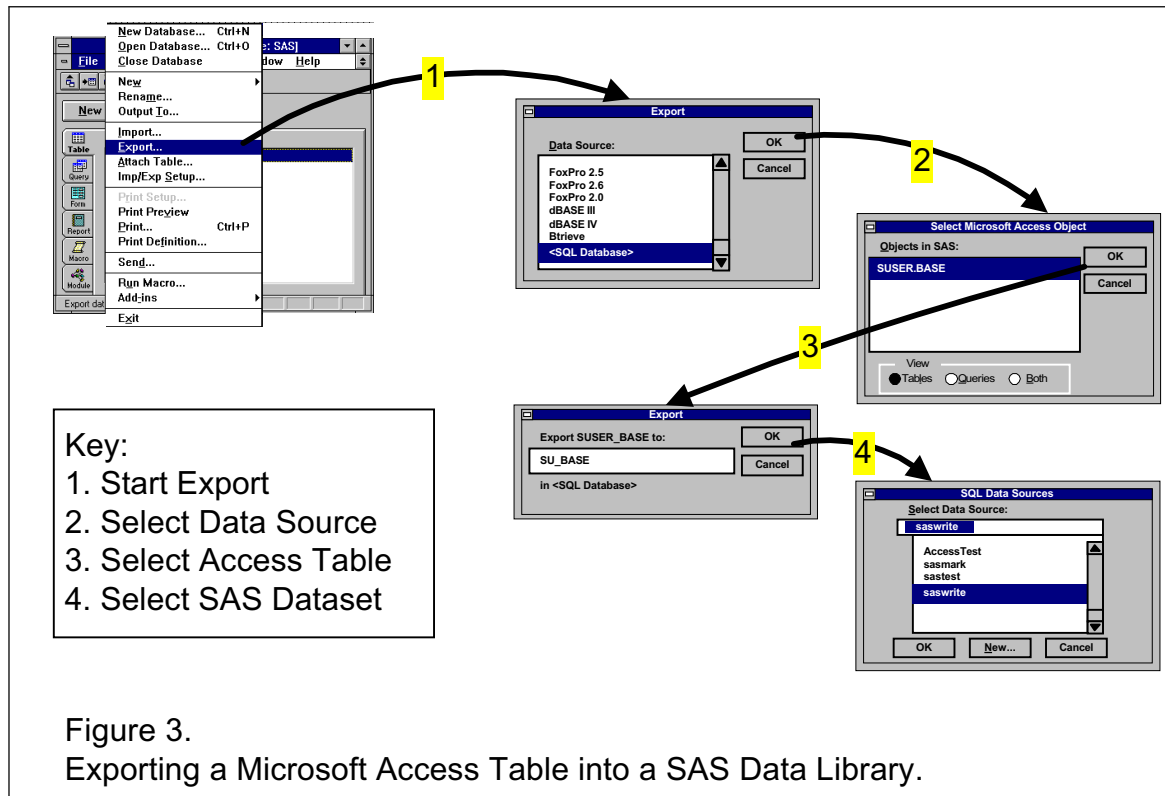


Figure 2.
Importing a SAS Dataset into Microsoft Access.

The Import procedure is as follows (see Figure 2):

- Click File, Import...
- Select a Import Data Source, ie. '<SQL Database>'.
- Click OK.
- Select Data Source, eg. 'sasuser'.
- Click OK.
- Select Tables in <SQL Database>, eg. 'SUSER.BASE'.
- Click Import.
- In the Information window, click OK.
- Data is copied into a new Access table, ie. 'SUSER_BASE'.



The Export procedure is as follows (see Figure 3):

- Click File, Export...
- Select a Export Data Source, ie. '<SQL Database>'.
- Click OK.
- Select Microsoft Access Object, eg. 'SUSER_BASE'.
- Click OK.
- Select Table to Export to in <SQL Database>, eg. 'SU_BASE' (see Problems below).
- Click OK.
- Select Data Source, eg. 'saswrite'.
- Click OK.
- Data is copied to SAS dataset, ie. 'USER.SU_BASE'.

2.2.1 Problems

There are several problems when Exporting Microsoft Access Tables to SAS Data Libraries:

- SAS Datasets can only be defined from Microsoft Access as single level names, so a USER libname must be allocated in the ODBC setup to receive the Exported Table.
- SAS Dataset names can only be a maximum of 8 characters. The names cannot include punctuation or blanks. They can only include alphabetic characters (ie. A-Z or a-z) or

underscore as the first and subsequent characters. Numeric characters (ie. 0-9) can only be used as the second and subsequent characters.

- SAS column names have the same rules as SAS Dataset names. Microsoft Access has a menu option (ie. File, Imp/Exp Setup...) to allow changes to be made to the Field Information before Exporting.

2.3 Microsoft Excel 5.0

Microsoft Excel can only use the SAS ODBC Driver to read from SAS Data Libraries with the Get External Data menu option.

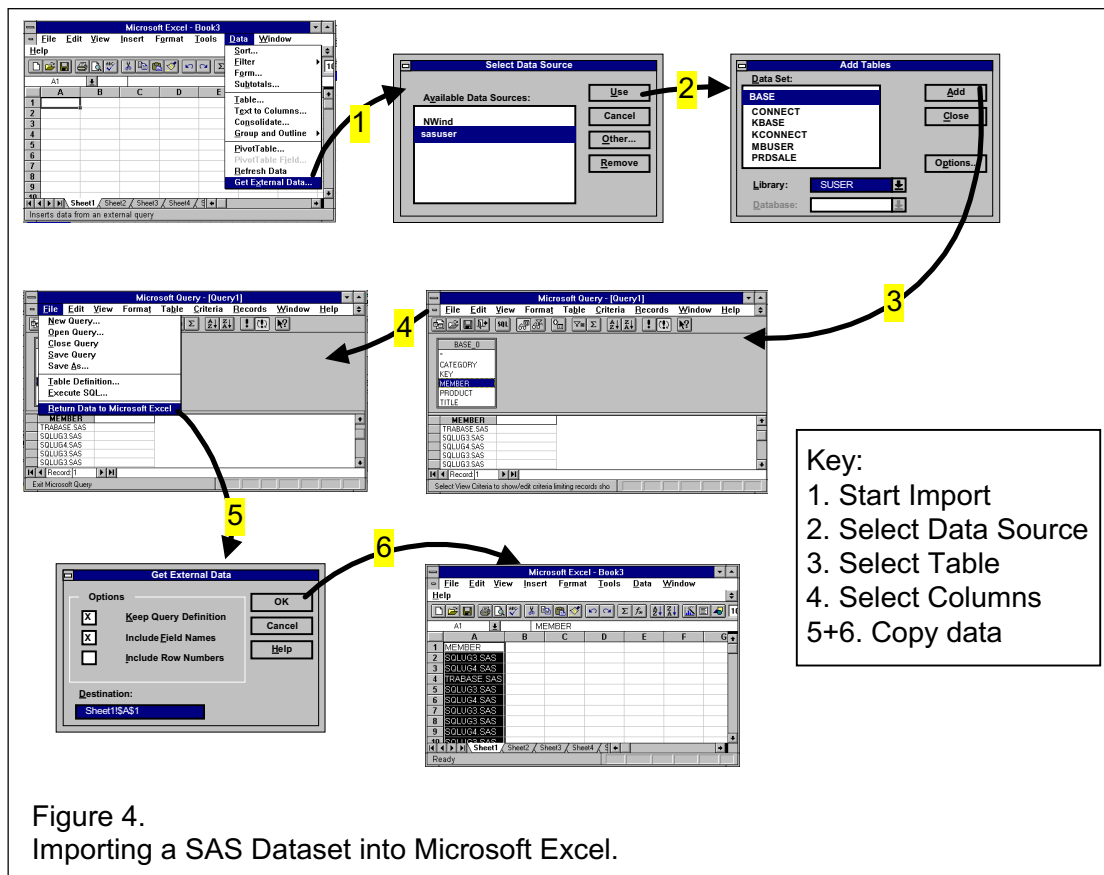


Figure 4. Importing a SAS Dataset into Microsoft Excel.

The procedure is as follows (see Figure 4):

- Click Data, Get External Data...
- Select a Data Source, eg. 'sasuser'.
- Click Use.
- Select Library, eg. 'SUSER'.
- Select Data Set, eg. 'BASE'.
- Click Add.
- In the Query window, click all the field names required from the field list, eg. 'MEMBER'.
- Click File, Return Data to Microsoft Excel.
- Select Destination.
- Click OK.
- Data is inserted at the destination location.

2.4 Visual Basic 4.0

Visual Basic can use the SAS ODBC Driver to read from SAS Data Libraries, with shared read-only access, using the following DatabaseName and Connect strings, eg.:

Set DB = OpenDatabase("sasuser", False, True, "ODBC;")

Visual Basic can use the SAS ODBC Driver to write to SAS Data Libraries, with exclusive update access, using the following DatabaseName and Connect strings, eg.:

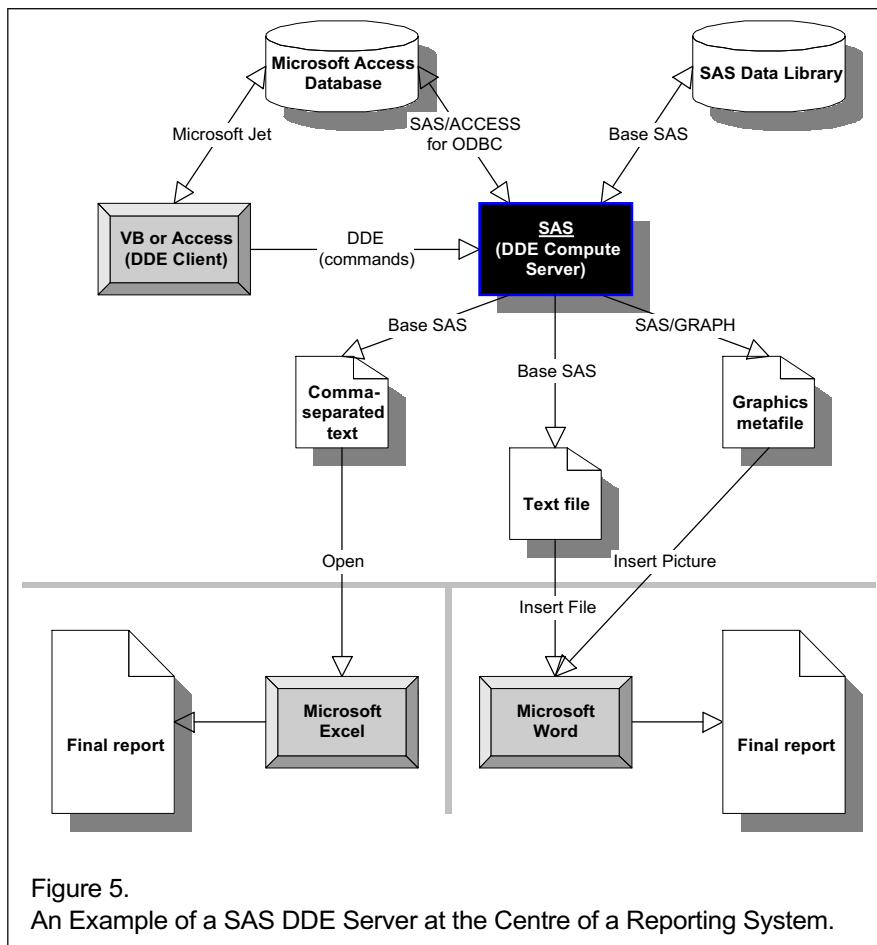
Set DB = OpenDatabase("saswrite", True, False, "ODBC;")

The database, 'DB', can be manipulated in the same way as any Microsoft Access database can be used, using standard Visual Basic Data Access functions and methods. The SAS Datasets within the ODBC Data Source can be accessed using the standard SAS 'libref.member' naming conventions, via Jet SQL supplied as part of Visual Basic and Microsoft Access.

3. DDE (Dynamic Data Exchange)

In the same way that SAS can be used to start and access Windows applications, SAS can be accessed by other Windows applications using DDE operations. The DDE interface allows data to be sent to SAS via code (eg. %LET statements, DATAstep code with instream data, etc.). It is possible for SAS Display Manager commands to be executed, menu items to be clicked, SAS code to be written into the Program Editor, in fact any SAS operations that can be performed using the keyboard, using SendKeys instructions with include special string values for special keys, eg. [ALT], [CNTL], [ESC], [ENTER], [F1], etc.

Data cannot easily be passed back to the calling program via DDE, but can be written to a file which can be read by the calling application using SAS code, eg. as an Microsoft Access Table using SAS/ACCESS for ODBC, as comma-separated text to text file, or as a Graphics Metafile (eg. *.CGM, *.WMF) using SAS/GRAPH®, etc. (see Figure 5).



3.1 Visual Basic 4.0

Allows any Visual Basic application to start SAS using a Windows Shell function. When the SAS session has been initialised, commands can be sent to SAS using the SendKeys function, eg.:

```
rc = Shell("d:\sas.exe", vbMaximizedFocus) 'start SAS in maximized window
For i = 1 to 1000      'wait for SAS to initialize
    DoEvents()
Next i
SendKeys "%GP", True      'Globals, Program Editor
SendKeys "{%}let macrovar = 1234;~", True 'set SAS macro variable
SendKeys "dm 'af c=lib.cat.prog.frame' af;~", True 'run AF application
SendKeys "%LS"          'Locals, Submit
SendKeys "%GP", True      'Globals, Program Editor
SendKeys "endsas;~", True 'Exit SAS
SendKeys "%LS"          'Locals, Submit
```

Additional processing by the Visual Basic application would be required if the SAS part of the application allows user input. The Visual Basic application must be hidden from view while the SAS part is executing. One way of performing this is to have a regularly executed function which initially creates a temporary 'locking' file, and then hides the Visual Basic window while this file exists. The SAS part can then be seen and run as required. When the SAS processing has been completed, SAS can delete the 'locking' file to tell Visual Basic to restore the visibility of the calling window.

4. Conclusions

This papers has only scraped the surface of what is possible using SAS as a file server or compute server for other Windows applications. It should now be clear that a large number of different PC users could benefit from using SAS effectively as a 'black box' processor with their own applications, reducing the need to fully train them in SAS coding techniques. The SAS Data Libraries and SAS Application development can be done for them by SAS specialists, providing the users with a well documented and stable interface that they can use without any requirement for prior knowledge of SAS.

5. References and further reading

- SAS ODBC Driver Technical Report: User's Guide and Programmer's Reference, Release 6.11
- Microsoft Visual Basic: Language Reference, Version 4.0

The author is a consultant for Holland Numerics Ltd and can be contacted at the following address:

Philip R Holland
Holland Numerics Ltd
94 Green Drift
Royston
Herts. SG8 5BT
UK
e-mail: <phil.holland@bcs.org.uk>
tel. (mobile): +44-(0)7714-279085

*SAS, SAS/ACCESS, SAS/SHARE, SAS/SHARE*NET and SAS/GRAPH are a registered trademarks of SAS Institute Inc., Cary, NC, USA.*