

The SAS System and Data Exchange

Clarence Wm. Jackson
ECI, Information Services Dept
City of Dallas

Abstract

The purpose of this paper is to describe one use of the SAS System as a Data Exchange tool at the City of Dallas. For this paper, Data Exchange defined as the process of moving digitized data from one computer platform or application to another. This paper will briefly review some of the reasons data exchanges between Mainframe and Personal Computer applications are needed, and some tools in use today to facilitate data exchanges. Data transfer tools between the PC and mainframe, OLE (object linking and/or embedding), DDE (Dynamic Data Exchange), and ODBC (Open Data Base Connectivity) will be discussed. Also, an example using a SAS server application under OS/2 DDE link to create a spreadsheet in a Lotus 123 for Windows client application will be reviewed.

Introduction

The majority of all mission critical detail data for the **City of Dallas** is stored on the mainframe in either **VSAM**, **AMIGOS**, or **DATA COM/DB**, but the majority of user department analysis of this data is summarized and accomplished on personal computer workstations. Numbers from the mainframe are crunched and many times, are re-entered into PC applications for reports to management as **Lotus 123** and **Excel** spreadsheets, **Word** or **WordPerfect** documents, **dBASE**, **Microsoft ACCESS**, **Q&A** and **PARADOX** databases, **Lotus Notes**, or what ever other PC tools that are available to the department.

Many batches of printed reports of detailed data and totals are reviewed only to retrieve a few numbers for further analysis and summarization in a spreadsheet. The numbers are highlighted, marked, or otherwise noted for manual entry into the spreadsheet. Once entered, the data is summarized, then copy/paste into a word

processor for some type of report, or combined with numbers from other printed reports into other spreadsheets. Care is needed in not missing any number noted, and in putting in the correct numbers. Also, finding the incorrect entry is time consuming.

Manual process of this type, reviewing many mainframe reports and re-entering the data into other PC applications, is a first step low level method of client/server and data exchange. Cutting and pasting data within the PC environment is another method of data exchange, but it's not possible to cut and paste from the mainframe, so we download or upload the data if we can do it without much hassle.

The chances of error in this process are great, and the error rates much higher than if the process was somehow automated to move the downloaded or printed data directly into the PC application. There are ways to automate these processes, but you need the right tools to do it. SAS is not the total answer, but it can be a part of a total solution in automating data exchanges between the mainframe and PC, and also between different PC applications.

Tools to facilitate Up and Downloads

Tools available for data exchange vary with the operating systems be used. On the PC side, WS/LAN is available for Windows, while RUMBA is the tool used on OS/2 to download the data from the mainframe to the PC workstation. And then there's SAS/CONNECT with it's Remote Library Services that will allow you to run mainframe SAS sessions from your PC. SAS/CONNECT is not available at the City of Dallas at this time, and will not be discussed.

Both of these "gateway" tools offer character set

translations for ASCII on PC's and EBCDIC for the mainframe, and as long as your data is "clean" (no packed data or proprietary graphics characters), your uploads and downloads can save you lots of time in exchanging data between the mainframe files and your PC. You can use SAS to unpack and translate the data to character and numeric on the mainframe host and write the data to a PDS or QSAM file with a block size of 8000 or multiple of 8000, then use RUMBA or WS/LAN to download the data to your PC as ASCII with carriage returns and line feeds (CR/LF) options.

If you choose to transfer the file as a SAS dataset transport file using PROC CPORT and will be reading it on the other end with PROC CIMPORT, use the same options as above, but do not include the carriage returns and line feeds option. Refer to the SAS Procedures Guide, v6, 1st edition for details on setting up SAS transport files.

Once you get your data exchanged between platforms, you then need to prepare it for your application's format. Different applications require data to look a certain way, such as dBASE files, comma delimited and/or tab delimited files, word processing and spreadsheets.

You can use SAS on the mainframe to prepare your data prior to downloading, so that the exchange between platforms will be cleaner. This reduces the chances of corrupting your data once exchanged, making it cleaner and reliable for your PC application without much manual work on your part.

Keep in mind that the file size capacity on the mainframe and PC are different, so you don't want to download more data than your workstation can accommodate. Be careful! Before you exchange data between platforms, determine your space requirements. Also, you can use SAS to summarize your detail data, and pass the summarized data to your workstation for further analysis. Let the mainframe (server) take care of the work, and pass only the data

you want to work with back to your PC (client).

Now, to automate getting that data into your other data presentation applications. SAS on the PC side, using the same syntax as the mainframe, plus the facilities that are available on the PC platforms, will allow you to further automate the process of data exchange by acting as a server to your applications (clients). Using DDE, OLE, and/or ODBC, SAS on the PC can be used to exchange your data with your other applications.

- **OLE (object linking and/or embedding)**
requires linking and/or duplications of data, but it's easy to do. Requires lots of resource on the PC because of embedded objects in clients. (Windows and Mac only)
- **DDE (Dynamic Data Exchange)**
links can be activated/deactivated as required. Client applications are manipulated by the server application. Requires both applications to be active, but the server application can start the client. Server "talks" to the client in the client's syntax. (Windows and OS/2 only)
- **ODBC (Open Data Base Connectivity)**
no links, and the application that created the data does not have to be active. Files are read directly using the ODBC drivers, and all file structures can be read with SQL statements through the ODBC Manager. The data is returned in the format of the application you are currently using. This allows a Lotus 123 application to read a SAS dataset that was either created on the mainframe or PC. However, the ODBC drivers are needed and are supplied with the software, like printer drivers are supplied with the printers. But being able to read any file created on multiple applications in a single applications simply by changing the driver is nice indeed. DataLens for Lotus Windows

products uses ODBC thru the "Connect to data" menu options. (Windows only)

So we're talking about a client/server relationships between the mainframe and PC, and also between PC applications. And a little data warehousing is involved, but mostly data exchange.

Client/Server and Data Warehousing?

Data Exchanging is essential in client/server systems and data warehousing. There is a lot of information out there on client/server systems, and data warehousing essential to setting up your client/server systems. We are doing this type of work now, but we may not call it "data warehouse" or "client/server". We may call it "archiving and purging", or "give me a report like this from the mainframe data".

Data warehousing involves many things, but especially the organization, storage, and assessability of data as information. There are many books available on this topic, as it's too much to cover in this paper. Think of it as the optimum balance in data storage and upkeep and information access of operational data. Bill Inman, is the "Father" of Data Warehouse as a principle and discipline of object orientated organization, has many such books available.

The terms are meant to focus your attention on the ideas of data organization around the functions of the organizational units, while optimizing the information available from the data. Downsizing at the City will not eliminate the data that's stored, but we can organize our data the way we use it. Understanding our data, how it is being used, updated, stored, etc, goes a long way toward understanding client/server and data warehousing, and data exchanges between the warehouse server and server to the client. According to many today, mainframes makes the best data warehouses.

So what does all of this have to do with exchanging data? (It's coming!) Simply,

client/server systems grew from the need to have clean, automated data exchanges from the server (holder of the data) to the client (user of the data) through proper data warehousing structures. Client/Server is no good without data warehousing, and data warehousing is no good without exchanging data. There needs to be reliable ways to exchange data.

Consider that most of the data is being stored (and warehoused) on the mainframe disks and especially tapes, and will not change much after it's put to the files. Payments, mail notifications, tickets, budget and financial data (especially for prior years) do not change once it's been put to the files. This data is considered "historical", but if it's needed in reports, it could be considered "managerial". Current mission critical data would be considered "operational". Operational data is usually the only data that needs to be online, but a lot of data is kept online because of the reports needed for analysis.

For instance, prior years' budget expenditures and payments of all types should not change, and can be archived, except it may be needed online to compare to current information. Although the resources being used to house the inactive data with the current data raises the cost of the system, the cost is justified by the information available. However, those items will not change, so the sum totals and other statistical information of those items will not change. And if you only need certain information of item groups, why keep all the detail items?

So we archive (warehouse) the inactive detail data to tape, but if we need to do analysis from it, we must read it all in each time it's needed. A Data Warehouse properly designed would provide the totals without needing to read the data each time. Summary reports of archived data also provide the same information, and is usually the information that is entered into the PC application. But a request for different analysis of the data could cause the summary reports to be inadequate, and results of more reads from the warehoused data.

In many ways, we are doing data warehousing and treating the mainframe as a server for our client information. But we can do it better. The need to download data from the mainframe to PC's to enter into a spreadsheet is a step up and improvement from printing the data on a report from the mainframe to enter into the spreadsheet. Downloads are done to bring the data closer, to review and analyze. Client/server systems grew from the needs of users to have access to their data. There are many spreadsheets around that just report the summarized data on various detailed items taken from the mainframe, and that data is usually manually entered, maintained, and verified.

So what if someone transposes a number from a Resource report on their budget or other spreadsheet? Which number will be trusted? How long would it take to find the problem?

Clean, automated data exchanges would reduce the probabilities of errors, so the rise in requests for client/server systems. Data exchanges between platforms and applications are the key to client/server systems. But, most client/server systems started with downloading and uploading simple data files, then set the standards to automated the processes to reduce errors.

A Sample Application

The sample process will demonstrate data exchanges in a simulated client/server system. Using Resource data stored on VSAM files processed in SAS V6.07 on our MVS/ESA mainframe, several versions of data sharing is presented of summarized data in report form, downloaded as ASCII, and processed in SAS V6.08 on OS/2, and passed to a Windows Lotus 123 (release 4) spreadsheet running in WIN/OS2 through a DDE Link. This data could also have been downloaded as a SAS transport file using PROC CPORT/PROC CIMPORT.

I would have loved to use ODBC, which would allow Lotus 123 to read the data from the native PC SAS dataset format, but ODBC is only

available in Windows, as is OLE 2.0. ODBC would allow SQL queries to be passed from Lotus 123 through the ODBC drivers, and passed the data back as requested in the closest data format available. But, we have OS/2 currently, and that's fine. SAS uses whatever is available on whatever operating system it's running on to exchange data. OLE (object linking and/or embedding), DDE (Dynamic Data Exchange), and ODBC (Open Data Base Connectivity) are supported on Windows only. DDE is the only one of the 'good' exchange methods available on OS/2.

Since we don't have Lotus 123 for OS/2, but we do have a Windows version, and OS/2 can run Windows (slowly!), Lotus 123 runs in Win/OS2, and allowed to run prior to executing the SAS job. I begin by opening the client application, in this case Lotus 123 for Windows, but I don't have to load the worksheet. If both were native OS/2 applications, I could start one with the other. This can be done with the current configuration in place, but for this paper, I will not discuss.

Once I start the application, I can use SAS to pass DDE statements through the DDE Link. The code (see exhibit 1) run on a PC will read the downloaded Resource data, manipulate it, call up a Lotus 123 spreadsheet template, put some data to the spreadsheet template, and save the file as an updated spreadsheet..

For Lotus, and valid command, macro, range name, "Classic 123" statements, etc, can be passed to through the DDE Link, as well as the data that is being processed through "PUT" statements after a "FILENAME {fileref} DDE ..." statement that establishes a SYSTEM connection link between them. This process will work with any DDE compliant OS/2 or Windows application, not just Lotus 123.

Benefits and Trade Offs

There must be some benefit to automating data exchanges, other than reducing errors, saving time, cutting down on paper dust, and using less

keystrokes to enter your data.

Maybe it could improve your production, but that will depend on the trade offs you would need to make, such as getting SAS on your PC, learning about DDE, OLE, and ODBC data exchange methods.

Then again, ask your department's budget analyst the next time they're away from their PCs what they think about it. And if you find yourself POKEing (a DDE term, find out what it means), don't worry, you could either be benefiting or making a trade off. You decide.

CONCLUSION

Data Exchanges are fundamental to doing our work as City of Dallas computer professionals. Others rely on the data's content for operational processes, managerial analysis and reporting, and historical and trend reporting, etc. Many methods other than manually exchanging data are available, and SAS offers only one of many solutions available.

If I were to rate the three methods of exchanging data within the PC environment, I would go with Windows so that I would have all three methods available for use. So, in reverse order of usability in automation, they are:

- **OLE (object linking and/or embedding)**
- **DDE (Dynamic Data Exchange)**
- **ODBC (Open Data Base Connectivity)**

REFERENCES

SAS Companion for the IBM OS/2
Environment (Version 6, 1st edition) SAS
Companion for the Microsoft Windows
Environment (Version 6, 1st edition)
Lotus 123 for Windows Usage Guide,
release 4.0
OS/2 Users Guide, Ver 2.0

ACKNOWLEDGEMENTS

SAS, the SAS System, and other SAS products are registered trademarks of SAS Institute, Inc, in the USA and other countries.

ODBC, OLE 2.0, Word, Excel, Access, and Windows are registered trademarks of Microsoft Corporation.

Lotus Notes, DataLens and Lotus 123 are registered trademarks of Lotus Development Corporation and IBM in the USA and other countries.

OS/2 is a registered trademark of IBM in the USA and other countries.

All other products mentioned are registered trademarks either by product name or Company.

This document was written at home by author as a handout for the City of Dallas Computer Information Forum meeting on July 18, 1995. Please distribute freely. Comments are welcomed. You may contact me at 670-3704, or via EM as 'CJ'.

* **Exhibit 1 - SAS code to write to LOTUS 123 spreadsheet ;**

```
%macro sleeper (howlong);
```

```
data _null_;
  x=sleep(&howlong);
run;
```

```
%mend sleeper;
```

```
options missing=0;
```

```
filename instuff "c:\cjackson\ft-area\prk-data.dat" lrecl=177;
```

```
data resource;
  infile instuff pad missover;
inPUT @ 1 FY      2.
      @ 3 ORG     $4.
      @ 7 ORGNAME $char35.
      @ 42 ACCT   $4.
      @ 46 DES    $char22.
      @ 68 BAMT   zd14.2
      @ 82 OCT_AMT zd8.
      @ 90 NOV_AMT zd8.
      @ 98 DEC_AMT zd8.
      @106 JAN_AMT zd8.
      @114 FEB_AMT zd8.
      @122 MAR_AMT zd8.
      @130 APR_AMT zd8.
      @138 MAY_AMT zd8.
      @146 JUN_AMT zd8.
      @156 JUL_AMT zd8.
      @162 AUG_AMT zd8.
      @170 SEP_AMT zd8. ;
```

```
if des="" then des="No Description";
desc=quote(des);
spent=sum(of oct_amt--sep_amt);
if spent > bamt then do;
  overage=spent-bamt;
  to_spend=0;
  msg2u="Over Spent";
end;
else do;
  overage=0;
  to_spend=bamt-spent;
  msg2u="Money Left";
end;
run;
```

```
proc sort;
  by org acct;
run;
```

```

proc freq;
  tables msg2u;
run;

proc tabulate;
  class org msg2u;
  var to_spend overage;
  table org*msg2u all,
    to_spend*(sum*f=dollar14.2)
    overage*(sum*f=dollar14.2);
  keylabel n="total records" sum="total amount";
run;

options noxwait noxsync;

* create link with 123 windows app as "sas2app" ;
filename sas2app DDE '123W\System';

data _null_;
* send dde statement to 123 to retrieve file sas2dde1.wk4 on a drive ;

  file sas2app;
  put '[run(/fr{esc}{esc}a:\sas2dde1.wk4~)]';
run;

%sleeper (20);

* create link with the 123 file sas2dde1.wk4 in cells a3..r14 ;

filename sas123 dde '123W\A:\SAS2DDE1.WK4!A:A3..A:R14' notab;

data showit;
  file sas123;
  set resource;
  outdde+1;
  if outdde > 10 then do;
    put "Spreadsheet after DDE update";
    stop;
  end;
* write to the lotus 123 file using put statement;
  put org $ '09'x acct $ '09'x desc & $char22. '09'x
    OCT_AMT '09'x
    Nov_AMT '09'x
    Dec_AMT '09'x
    jan_AMT '09'x
    feb_AMT '09'x
    mar_AMT '09'x
    apr_AMT '09'x
    may_AMT '09'x
    jun_AMT '09'x
    jul_AMT '09'x
    aug_AMT '09'x
    sep_AMT '09'x

```

```

        bamt      '09'x
        to_spend  '09'x
        overage   '09'x;
run;

%sleeper (10);

data _null_;
file sas2app;
* save the file as "afterdde.wk4" ;
put '[run(/fs{esc}{esc}{esc}a:\afterdde.wk4~)]';
run;

%sleeper (60);

proc print n u;
run;

```

Org	Accl	description	october	november	december	january	february	march	april	may	june	july	august	september	budgeted	left to spend	overspent
-----	------	-------------	---------	----------	----------	---------	----------	-------	-------	-----	------	------	--------	-----------	----------	---------------	-----------

Org	Accl	description	october	november	december	january	february	march	april	may	june	july	august	september	budgeted	left to spend	overspent
5002	1101	SALARY/CIVILIAN"	24072	30797	27373	24217	45791	29937	33258	32428	0	0	0	0	461039	213168	0
5002	1201	OVERTIME"	237	193	141	0	2083	663	1228	818	0	0	0	0	5120	0	139
5002	1203	S.I.P. (1111)"	0	0	11064	0	0	0	0	0	0	0	0	0	11158	94	0
5002	1207	No Description"	195	0	0	552	0	0	0	0	0	0	0	0	0	0	747
5002	1301	PENSIONS/CIVILIAN"	2410	2813	3775	2870	4412	2734	3127	2984	0	0	0	0	42193	17068	0
5002	1303	LIFE INSURANCE"	61	61	61	40	81	56	56	61	0	0	0	0	30394	29917	0
5002	1304	HEALTH INSURANCE"	2277	2277	2277	1518	3036	2088	2088	2277	0	0	0	0	0	0	17838
5002	1306	FICA"	132	149	159	157	237	154	169	157	0	0	0	0	1568	254	0
5002	1401	SICK LEAVE CIVILIA"	209	817	204	1046	568	194	734	76	0	0	0	0	0	0	3848
5002	1402	VACATION LEAVE CIV"	3639	1288	2311	2970	1803	1467	1418	1788	0	0	0	0	0	0	16684
Spreadsheet after DDE update																	