Using DataLoad to Streamline Data Entry, Maintenance, and Setup.

John Hebert

Waukesha County

Introduction — If you use a computer (whether you're an end user or an IT pro) the odds are that you've experienced the tedium of repetitive data entry or data maintenance. If that data is (or could be) available in a spreadsheet, then it can be staged in DataLoad's spreadsheet style interface and read into your Oracle form just as if it was keyed (only a lot faster…) If you can get your data into a spreadsheet, this tool is worth a look.

The PowerPoint presentation associated with this white paper provides actual examples of using DataLoad and presumes a basic understanding of Oracle applications and Microsoft Excel. This white paper provides an overview of DataLoad's two versions and some application considerations/solutions beyond the product's installation instructions. The solutions presume familiarity with application activities related to defining and setting up concurrent request sets.

Problem Statement — Our installation of Oracle (Government) Financials is set up to include nine accounting segments (for 55 potential keystrokes) and even though skilled clerical staff know their way around a keyboard, mistakes happen. Managers end up spending a fair amount of time each year searching out keying errors where the charge is to valid but incorrect account code combinations. Furthermore, we had some data entry bottlenecks where the data source was already contained in or compiled into spreadsheets.

When I was asked what could be done to alleviate the bottlenecks, my options were limited. Budgets for units of local government tend to be very lean and our application module support is provided by a team of one – both money and time are at a premium. Previously, I had great success using a tool called DataLoad to assist me during upgrades and conversions. I was familiar with the tool, could provide a demo on short notice, and knew it was within our limited budget so I presented the two available options.

DataLoad Classic — Once downloaded and installed, using DataLoad is relatively simple - you define a 'Macro' in the spreadsheet style interface. The 'Macro' maps and automates your form control a data entry actions in the form– it replays the positioning, keystroke sequences, and copies the data from the spreadsheet and applies them the form.

DataLoad Classic isn't limited to Oracle Applications or Oracle Forms. It is a generic tool that can be used to enter or maintain data in many business and personal applications. If the application presents a form on your screen and your data is in a spreadsheet, there is a good chance you can use DataLoad to automate your entry/maintenance tasks.

Film clips in the power point presentation associated with this paper include numerous examples using DataLoad Classic to run macros. Please refer to that presentation as well as the vendor's documentation for information on running DataLoad Classic.

As far as DataLoad Classic is concerned, you can't beat the price; at of the time of this writing it is a free download (a big thank you to JD Stuart Ltd. and Edenbrook!) You can find it at *http://www.dataload.net/downloads/index.html*

DataLoad Professional — Specifically targets Oracle Applications. It gives Oracle Application users the ability record and play DataLoad files. Using the record feature eliminates the need to manually transcribe your keystroke steps into a DataLoad spreadsheet and the playback functionality is significantly quicker than a corresponding Macro load – this is a real plus if you're trying to process large volumes of records.

DataLoad Professional's added features come with a 'price' - it requires some setup work and the Professional version isn't free; you'll need appropriate licensing. The vendor's online manual at http://www.dataload.net/help/recplay/setup.htm explains needed setup for use with various versions of Oracle Applications.

Film clips in the power point presentation associated with this paper include some examples using DataLoad Professional. Please refer to that presentation as well as the vendor's documentation for additional information on setup and running DataLoad Professional.

Once you've installed the professional version and setup up 'Record' and 'Play' user ID's you're set to go – almost. You still need to determine how you will transfer your DataLoad files to your Oracle Applications. The remainder of this paper illustrates how we handle the data transfers from windows desktops and our AIX servers.

Waukesha County's data transfer solution — The Macro version is a great tool and our power users could quickly grasp it; they would be off and running in no time without any need for IT assistance. The Professional version required some IT effort to setup and implement, it runs considerably faster, and produces an activity logs. The Professional version was our choice - we weren't targeting power users and we would have to support the implemented solution.

One of the 'problems' we encountered in 11.5.10 came straight out of the setup instructions: "Oracle Applications systems at this software level will only record/play FLD files in a single directory and by default that is the following location on the application server: $COMMON_TOP/admin/log/[Context Name]." We were reluctant to open this area to our users for ftp access. The Inbound directory was already defined in Init.ora (Util_File_Dir) as being accessible and was already being used by other interfaces. For consistency, we wanted the DataLoad file to go to Inbound so we opted for a two step process that involved moving the data to Inbound and then moving the data to $COMMON_TOP/admin/log/[Context Name].

Since the data is created on the user's desktop we set up the following batch file to temporarily map a drive and copy the file to the Inbound folder:

```
net use R: \\financialsprd\inbound
del  R:\Aging_Invoices.fld
copy C:\Work_Financials\Aging_Invoices.fld R:\
net use R: /delete
pause
```

A shortcut to run the batch file was placed on the user's desktop.

We then set up a couple scripts (AIX permissions set at 755) that are submitted as a Concurrent Request Set to process the files that the user placed in the Inbound folder. The first script copies the file from the Inbound folder to $COMMON_TOP/admin/log/[Context Name] while creating a log file that either contains a list of records processed or an error message:

```
# This Shell Script is for Aging Invoice DataLoad Interface
# Copy data from Inbound.
# i.e. $IN_TOP directory after the files have been loaded
if [ -f $IN_TOP/Aging_Invoices.fld ] ;
then cp $IN_TOP/Aging_Invoices.fld $COMMON_TOP/admin/log/FINDEV_bastst/;
egrep "VENDOR_NAME|INVOICE_AMOUNT_DSP" Aging_Invoices.fld;
else echo Aging_Invoices.fld File Not Found in $IN_TOP ;
exit 1;
```

```
fi;
```

The second script renames the file in the Inbound folder by appending a date stamp – this not only prevents an old file from inadvertently being copied a second time, but it also provides documentation of what was actually loaded and moved in the event problems are reported:

```
# This Shell Script for Aging Invoice DataLoad Interface rename
# Rename the dataload files present in $IN_TOP directory
# after the data has been copied
FILEEXTN=`date +%m_%d_%Y_%H_%M_%S`
if [ -f $IN_TOP/Aging_Invoices.fld ]
then
if [ -w $IN_TOP/Aging_Invoices.fld ]
then
mv $IN_TOP/Aging_Invoices.fld $IN_TOP/Aging_Invoices.fld_${FILEEXTN}
echo Aging_Invoices.fld File Renamed to Aging_Invoices.fld_${FILEEXTN}
fi
else
echo Aging_Invoices.fld File Not Found in $IN_TOP;
exit 1;
fi
```

Once the scripts were prepared and saved in our $CUSTOM_TOP we used Oracle Application's standard functionality to set up a Concurrent Request Set that users could submit to move their data from Inbound. The setup includes defining Concurrent Program Executables using the Host execution mode, Concurrent Programs to run the executables, and a Request Set (which included both Concurrent Programs.) Finally, we reviewed our Request Groups to determine which group(s) should have access to the newly created request set and added it to the appropriate group(s) or, when necessary, a new Request Group was created. By defining our DataLoad interface in this manner our users got a look and feel that is similar to other existing interfaces.

Once this type of interface was in place, our users were able to use DataLoad Professional independent of IT. Users simply verify the successful completion of the concurrent request before logging into a PLAY ID (as explained in DataLoad's documentation) to complete the load.

Lessons Learned — Our experience indicates that desktop events can adversely impact DataLoad. To avoid problems, our users are instructed to close any software that generates a pop-up message (Novell Notify, Microsoft Outlook, Desktop Sidebar, various messenger programs…) This recommendation was formulated when we first started using DataLoad (a couple versions ago) in a Novell environment where we noted that GroupWise Notify pop-ups presented a problem – they crashed our loads.

While preparing the film clips for the power point presentation associated with this paper, I failed to follow our own guidelines and discovered that in at our current version of DataLoad, in a Microsoft environment, Desktop Sidebar pop-ups didn't cause a problem. However until our migration to Microsoft is complete and I've had a chance to test various pop-ups during both Macro and Playback loads, we will continue the current recommendation to disable pop-up generating software for the duration of a load.

Along similar lines, beware of scheduled (or unscheduled) desktop software updates – pushed updates can ruin a perfectly good load -- a reboot message popping up in the middle of a load can cause extreme aggravation.

If you encounter problems running Macro loads, the best advice I can offer is to think rhythmically - use macro functions to mimic the data entry rhythm. For example, if, when manually entering the data you see a forms processing pause (such as slight delay after entering a query, saving a record, or completing fill from a LOV) then accommodate that pause in your load. Inserting strategically placed *SL() columns in your macro allows you to see and tweak the rhythm. It is better to err on the side of a slower load than have it fail.

You have alternatives to control timing. For some commands, you can go to DataLoad's preconfigured delays and modify the associated default settings. Appropriate changes to the default delays can minimize the need to insert *SL() commands, however, you loose the visual cue for the rhythm that the *SL() command provides. Furthermore, if you use DataLoad Classis for multiple applications setting default delays may not be the best answers since necessary delays may not be consistent across different applications.

We try to avoid doing Macro loads during scheduled background processes such as disk defrags or spy-ware sweeps. These processes can adversely impact desktop performance and result in the failure of a perfectly good load due to changes in response time. Likewise, avoid running loads during periods of impaired application response – one example might be year-end where we noted performance hits during some combinations of year-end reporting/tasks.

Consider breaking large blocks of work into more manageable clumps this particularly important if you've set up a load where you are doing a manual save after completion – the longer the load, the more you can loose and have to redo.

Finally, if your Oracle Applications are plagued with FRM-92100 disconnect errors you probably want to get them under control before deploying a DataLoad solution. Having to log back in because of a FRM-92100 disconnect is frustrating enough without having to review your load and recreate it minus the processed pieces.

Conclusion — The strategies that we develop for addressing the everyday business issues depend on numerous factors including the availability of and our exposure to various alternatives. At one end of the spectrum direct SQL updates can be quick and effective but Oracle support tends to frown on them and since most of us would prefer to avoid support nullification issues we tend to avoid direct updates. At the other end something like HP's (formerly Mercury Interactive) WinRunner gets good reviews from colleagues who are fortunate enough to have access to it. It very well may be terrific, but the last time we checked it was beyond any budget we've had (or will have.)

From my perspective, DataLoad is an affordable product that provides an interesting, simple, and functional option for getting data into and manipulating data in Oracle Applications (or just about any other form style application you use.) Just add a little patience and creativity and it will help get the job done.