Paper AS07

Analysis of Pharmacokinetic Data : Using SAS to Create an Autonomous Environment for the Pharmacokineticist

Steve Prust, Covance, Leeds, UK

ABSTRACT

When analysing pharmacokinetic (PK) data it is necessary to combine and handle various items of data (assay date/times, dosing date/times, concentration results, demographic data etc.). Typically, such data has to go through a series of SAS programming steps e.g. validation, merge, reporting of potential problems before it can be handled by the pharmacokineticist. A further aspect of this type of work is that the pharmacokineticist will be making decisions about data – e.g. which results are anomalous – that will require databasing prior to the production of tables, figures, and listings.

This paper looks at a setting where – in order to improve turnaround times and enable more autonomy for the PK department - a hybrid SAS® environment was created that enabled direct use of data by the pharmacokineticist together with automatic databasing of the pharmacokineticist's decisions, all without the need for SAS programming intervention

INTRODUCTION

The preparation of PK work – like all our work – is a co-operative effort between departments. However, unlike the mainstream trials data the PK data requires more complex interactions. These led to PK work taking a longer time than desired. We set about devising a new, faster method.

Another important factor was that –as in many technical situations – the "tail tended to wag the dog" – that is the infrastructure of the environment defines and controls the process. We wanted to reach a situation where the pharmacokineticist was more autonomous and able to respond quickly to changing business needs.

To effect a change we analysed the situation in terms of :

- what data was being handled
- what decisions and instructions were being made by the pharmacokineticist that affected data preparation
- the tools available,
- the range and variability of studies and data

This paper concentrates on how we made our changes, the outcome, and the lessons learned.

SCOPING THE PROBLEM

THE DATA

Typically the data required were :

- assay dates and times
- dosing dates and times
- demography
- randomisation
- body weight (for some studies)
- sample results

The sample results were usually available only some time after the rest of the data. The nature of PK assays meant there were frequently delays and/or complex queries. It was not uncommon therefore to have several transmissions of such data.

From this data a SAS dataset would be created containing sample profiles that the pharmacokineticist would use for input into their pharmacokinetic analysis software.

DECISIONS AND INSTRUCTIONS FROM THE PHARMACOKINETICIST

Every study is different and we realised there was no exhaustive list of what might be asked for. Those we anticipated were :

- identification of anomalous results
- profile start and end points
- profiles that were "intermediate" – that is would start with some drug already inside the body
- treatment of values below quantification level

We also needed to alert the pharmacokineticist to specific situations such as:

- Positive results prior to dosing
- Embedded values below level of quantification
- "late" positive values – i.e. after two or more values below quantification

TOOLS AVAILABLE

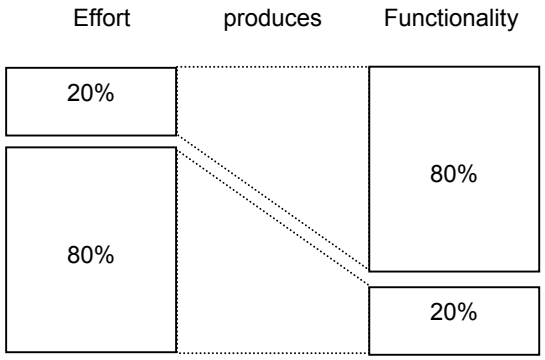
The tools available to us were principally SAS and Microsoft Office. There was some existing use of WinNonlin and some knowledge of the facilities available in S and SPSS. We had a preference for not introducing new software if possible because of costs (not necessarily purchase costs but the initial and ongoing costs of support and training).

In SAS we had made extensive use of SAS MACRO for process improvements and had made some use of SAS/AF principally in terms of utilities.

RANGE AND VARIABILITY OF STUDIES

Given that we are a contract research unit undertaking studies for many tens of diverse clients we had an instinctive view that we could not cover every single study we might encounter in future.

Adapting the Pareto principle to software development suggests a cost – results structure of :



We wanted to cater for all our studies and thus offer all our clients the same high-quality service but we were conscious that we could stray into the problem of over-elaboration in our solution. The likely cause of such variability in the system was perceived to be varying data structures and requirements. If we could cope with these in our system design we could hopefully keep build costs down.

SYSTEM DESIGN

WHAT TOOLS TO USE

The software tools decided upon were :

- The basic techniques of merging data were obviously to be done in SAS – we had the skills and already a lot of code
- the flagging of data by the pharmacokineticist was to be done using Excel with data being transferred from SAS to Excel and back. Excel is not a great tool – especially with SAS (precision of data can be lost, labels lost etc. etc.) but it was a familiar tool
- in moving SAS data to Excel and back we would have to build in some integrity checking for the data – PROC COMPARE has a very useful system return code facility that we could take advantage of
- to handle the variability in studies we would use a SAS/AF interface – allowing the pharmacokineticist to use SAS code flexibly but without needing to write code
- the whole system was to have a menu system using SAS/AF.

FLEXIBILITY – "CUSTOM CODE"

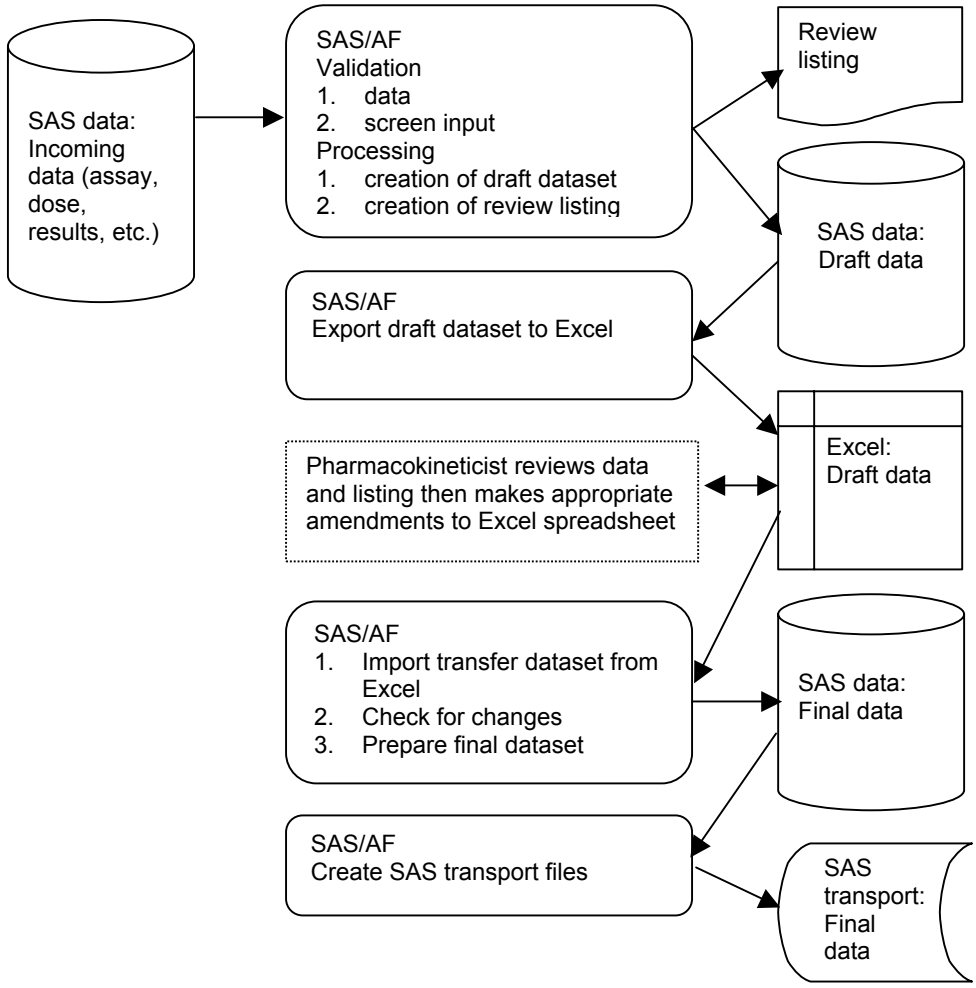
To cope with variable data structures a method of incorporating "custom code" was devised. The SAS code behind the AF screens would mostly be in macros. The method was to include the following macro code :

```
%macro macroname(macro parameters,custom=No);
<read in raw data>
data randxxx;
  set random dataset;
run;
...
%if %upcase(&custom) = YES %then %do;
  %include {custom code};
%end;
...
data rand;
```

```
  set randxxx;
run;
...
```

If custom code is required to cope with changing data specifications the SAS/AF code would invoke the macro with the parameter *Custom* set to "Yes". The appropriate code would be included by the macro. This code would take data from the non-standard specification and create regular style datasets (*randxxx* etc. in the above example).

FLOWCHART



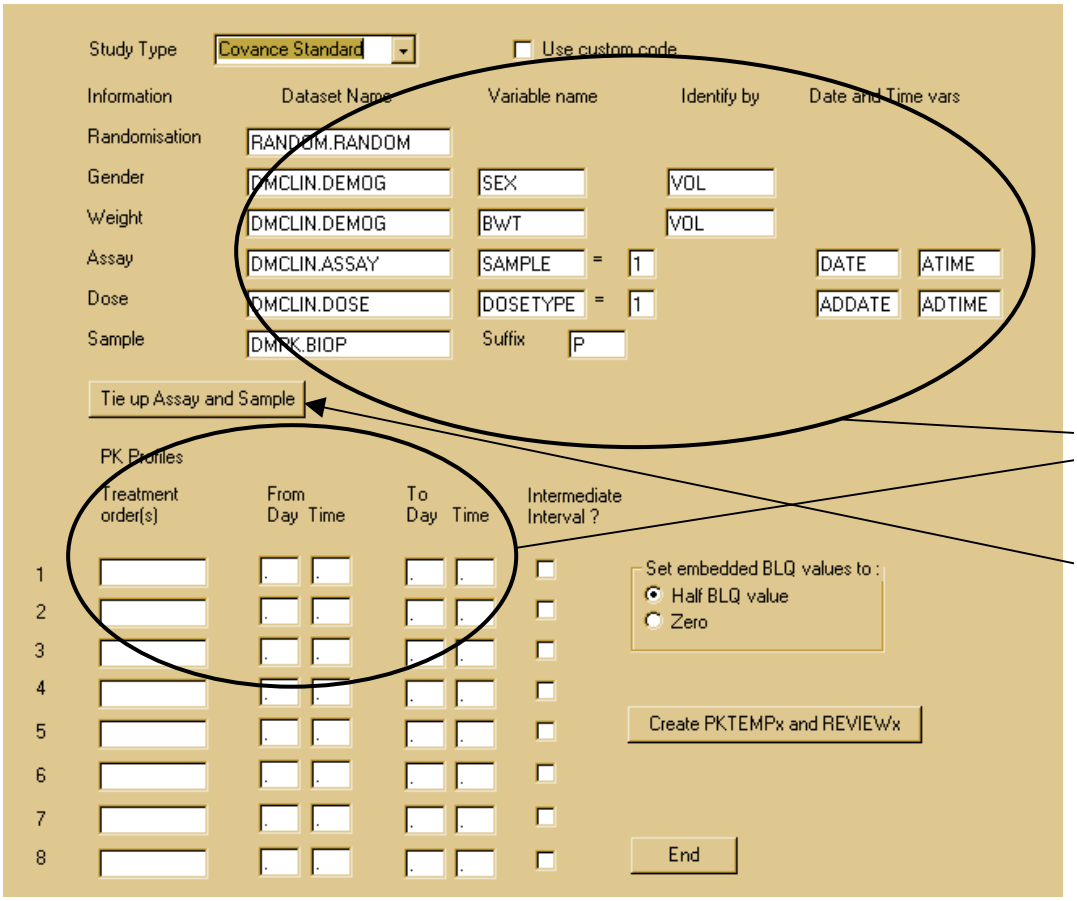
DATA CHECKING

ON SCREEN CHECKS

As a general principle all parameters entered on screen are checked for validity. The user of the system – the pharmacokineticist – has some knowledge of the data but is not expected to perform data checks themselves.

Additionally there is a facility to "tie-up" the assays taken and the results received. Normally one would expect standard data validation to deal with such issues during the data management phase of the project but given both the different time at which PK results can be supplied and the possibility of data being re-supplied (often in time pressure situations) this was seen as improving reliability and autonomy for the pharmacokineticist.

The screen shot below shows the screen areas checked by the system :



Typical of the AF code used for the above check is the following code fragment for the dose dataset information :

```
if not exist(teDosdsn.text) then do; /* Dose dataset does not exist */
  {error reporting code}
end;
else do;
  tableid=open(teDosdsn.text,'i');
  chknum=varnum(tableid,teDosDVar.text);
  if (chknum=0) then do; /* Dose date variable does not exist */
    {error reporting code}
  end;
  chknum=varnum(tableid,teDosTVar.text);
  if (chknum=0) then do; /* Dose time variable does not exist */
    {error reporting code}
  end;
  chknum=varnum(tableid,teDosvar.text);
  if (chknum=0) then do; /* dose var does not exist */
    {error reporting code}
  end;
  else do;
    cond = teDosvar.text || '=' || teDosval.text;
    rc=where(tableid,cond);
    if rc ^= 0 then do;  /* dose var and value are incompatible */
      {error reporting code}
    end;
    else if attrn(tableid,'nlobsf') = 0 then do; /* no values found for var=val */
      {error reporting code}
    end;
  end;
  rc=close(tableid);
end;
```

As this example perhaps shows AF code can use the same SAS facilities as SAS/MACRO and the DATA step. Particularly useful for such checking are the functions shown above of EXIST, OPEN, VARxxx, ATTRN, ATTRC

CHECKING REPORTS

Some checking results in so much information that it can't be shown on screen. Instead a report is produced – as is the case for the tie-up of assay and result data, which looks as follows :

Tie-up report between Assays (DMCLIN.ASSAY) and Results (DMPK.BIOP)

Analyte	Sub no.	Treat period	Day Day	Nominal time	Treatment description	DMPK.BIOP Result	Comment
XYZ	11	1	11	30.00	10 mg XYZ		No matching result
XYZ	12	1	11	24.00	10 mg XYZ	NR	No matching result
XYZ	15	1	7	0.00	15 mg XYZ	NR	No matching result
XYZ	16	1	7	0.00	15 mg XYZ	NR	No matching result
XYZ	23	1	11	30.00	25 mg XYZ		No matching result
XYZ	23	1	11	33.00	25 mg XYZ	3.54	No matching assay

INFORMATION AND REPORTS FOR THE PHARMACOKINETICIST

Apart from validation checks as shown above a review listing is produced for the pharmacokineticist detailing all the items found for review :

- Time deviation
- Embedded values below level of quantification
- "late" positives
- pre-dose measurements not below level of quantification
- missing sample dates / times

For example :

Items identified for review in study ABC

ANALYTE	VOL	PROF	DAY	NTIME	COMMENT	CONC	ACTUAL TIME
XYZ-1	3	1	1	0	Day 1 pre-dose not blq	4.26	-0.33
XYZ-1	4	2	7	3	Sample date missing	6.63	.
XYZ-1	5	1	1	0.5	Time deviation	17.79	1.2

Another listing is produced of all the changes, flags etc. set by the pharmacokineticist on the draft dataset. This then comprises part of the study documentation.

Lastly, any custom code used for a study is also formally reported (any such code has to go through independent quality control).

IMPLEMENTATION

Because of the nature of this system is had to undergo validation in order to prove its fitness for use. The principles we adopted for validation were :

- Run the system in parallel to existing methods for three months or until at least one of each study design (single dose, parallel, cross-over etc. etc.) had been tested (whichever took longer) and perform formal comparison of results.
- Use of previous complex studies to stress-test the system

From this we have created a testbed that we can use in future validations of the system

SAS TECHNIQUES

SAS MACRO

The number of profiles that pharmacokineticist can request is flexible. To cope with this we invoked a macro with a named parameter that expected the profiles in the form of a repeating block of "<treatment order> <start day> <start time> <end day> <end time> <intermediate interval flag> / " and then used macro code of the form :

```
%let i = 1;
  %let profile = %qscan(&profiles,&i,%str(,));
  %do %while (&profile ^= );
    %let proford  = %qscan(&profile,1,%str(/));
    %let rest = %qscan(&profile,2,%str(/));
    %let profsday = %qscan(&rest,1,%str( ));
    %let profstim = %qscan(&rest,2,%str( ));
      {etc}
    ...
    %let i = %eval (&i + 1);
    %let profile = %qscan(&profiles,&i,%str(,));
  %end;
```

not particularly elegant, but effective for our purposes.

PROC COMPARE

We used PROC COMPARE when the data was re-imported from Excel. This was for two reasons :

1. to ensure that the structure of the data had not changed (rows and columns added, changed, or deleted)
2. to identify what changes had been made by the pharmacokineticist.

PROC COMPARE has a useful return code facility that can be accessed via the global macro variable. The following code demonstrates this :

```
proc compare base=&base compare=&compare out=&rptdif outdif outnoequal nosummary;
  by order analyte vol period profile pkday pkntime;
  var flagdev flaganom winonlin pklevel plevel pactime pkactime;
run;

/* review results of compare */

%let comprc=&sysinfo;
%let comprc3 = %sysfunc(putn(&comprc,binary16.));
%let result = ;
%let result2 = ;
/* note - not interested in first six type of difference */
/*%if 0 ^= %substr(&comprc3,16,1) %then %put Data set labels differ;*/
/*%if 0 ^= %substr(&comprc3,15,1) %then %put Data set types differ;*/
/*%if 0 ^= %substr(&comprc3,14,1) %then %put Variable has different informat;*/
/*%if 0 ^= %substr(&comprc3,13,1) %then %put Variable has different format;*/
/*%if 0 ^= %substr(&comprc3,12,1) %then %put Variable has different length;*/
/*%if 0 ^= %substr(&comprc3,11,1) %then %put Variable has different label;*/
%if 0 ^= %substr(&comprc3,10,1) %then %let result2 = %str(&result2 Base data set
has observation not in comparison.);
%if 0 ^= %substr(&comprc3,9,1) %then %let result2 = %str(&result2 Comparison data
set has observation not in base.);
%if 0 ^= %substr(&comprc3,8,1) %then %let result2 = %str(&result2 Base data set has
BY group not in comparison.);
%if 0 ^= %substr(&comprc3,7,1) %then %let result2 = %str(&result2 Comparison data
set has BY group not in base.);
%if 0 ^= %substr(&comprc3,6,1) %then %let result2 = %str(&result2 Base data set has
variable not in comparison.);
%if 0 ^= %substr(&comprc3,5,1) %then %let result2 = %str(&result2 Comparison data
set has variable not in base.);
%if 0 ^= %substr(&comprc3,4,1) %then %let result  = %str(Changes in values
detected);
%if 0 ^= %substr(&comprc3,3,1) %then %let result2 = %str(&result2 Conflicting
variable types.);
%if 0 ^= %substr(&comprc3,2,1) %then %let result2 = %str(&result2 BY variables do
not match.);
%if 0 ^= %substr(&comprc3,1,1) %then %let result2 = %str(&result2 Fatal error:
comparison not done.);

%if &result ^= %then %do;
  %if &result2 ^= %then
    %let pkcomprc = %str(Changes in values detected AND other conditions - see LOG
and OUTPUT windows for full details);
  %else
    %let pkcomprc = &result;
%end;
%else
  %let pkcomprc = %str(No changes found - see LOG and OUTPUT windows for details);
```

REVIEW

When we started out we were not sure that we could design a system to meet our objectives, however by and large we are satisfied. The system works, it has made big savings on effort and elapsed time, it has enabled the pharmacokineticists to work in a far more autonomous way.

WHAT WORKED WELL FOR US :

- The overall design is robust and has been working well for some time. It was achieved without huge cost or a long build time.
- SAS/AF together with SAS MACRO is great for achieving flexibility – both in terms of validation and processing the data

WHAT WE MIGHT DO DIFFERENTLY :

- Maybe not use Excel – it isn't the best tool but it is well understood and widely known. At some stage we will move to SAS version 9 and I think we may look at other transfer mechanisms (e.g. XML) and reviewing methods for the pharmacokineticist.
- Also a better designed interface could be devised – at times the current interface is a bit "clunky"

IN THE FUTURE

- We have Oracle Clinical, CDISC, and CDISC-style datasets being used more and more. These present us with some synergies that we could take advantage of. They also potentially provide more variability – however we don't anticipate that it will necessitate changing the fundamentals of the current system design.

CONCLUSION

It is possible in SAS to build successful systems for the clinical trials environment. Key to the success in this project was keeping the system flexible - something which SAS/AF and SAS/MACRO are major tools for.

Another aspect of the system was making different software tools work together, this "mix and match" approach worked well (although Excel was not ideal it did have many virtues) and we will use this approach again when the occasion demands. It isn't necessary or indeed desirable to stay totally within the SAS environment.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at :

Steve Prust
Covance CRU
Apsley House
71 Wellington Street
Leeds LS1 2EQ
+(44) 113 237 3500
stephen.prust@covance .com