



Optimising a (supposedly) reusable program to reduce the need for manual changes

Pantaleo Nacci, Global Clinical Solutions,
BioPharmaceuticals Clinical Operations,
R&D, AstraZeneca, Cambridge, UK



Agenda

1. Introduction
2. Problem Statement
3. Initial Analysis
4. Implementation
5. Conclusions



1

Introduction



Introduction

- As described in the paper I presented last year in Belfast, AstraZeneca has been migrating all legacy A&R systems into a modern Statistical Computing Environment (SCE), entimICE-AZ, based on entimICE® DARE by Entimo® AG
- The latest (and last) legacy system to be ported, earlier this year, has been MedImmune's CARS, previously hosted in a SAS® Grid installation running SAS 9.4 on Linux™
- This system had been in use for several years by both CDM and SP, so that both functions needed support to timely and successfully migrate programming activities to the new system



2

Problem Statement



Problem Statement

- I helped the CDM staff with the porting of a standard program to create data set for JReview[®] consumption
- I worked on a restricted number of studies, but I soon noticed that
 - Some sections were (almost) always present in roughly the same order
 - Each study version had some specific sections or tweaks
 - A few study-specific macro variables were defined at the beginning, but paths were then mostly hardcoded
 - These issues required a substantial adaptation of the code before the actual porting could start



3

Initial Analysis



Initial Analysis

- No functional specification document was available
- No centrally maintained version of the program existed, so study programmers copied the latest one they knew about
- The naming convention adopted was `jreview_<project>_<study>.sas`
- The code looked very old, poorly commented, mostly ignoring available SAS functions, badly organised and thus very error prone
 - It relied heavily on visual inspection to identify which folders and data file types were present
 - It contained plenty of X statements and Linux commands, making it poorly portable
- There was not enough time for a proper porting exercise, so only minimal changes to make it work were applied, at least initially
- The macro variables and metadata automatically provided by entimICE-AZ would be very helpful to reduce the reliance on hardcoding in the ported program



4

Implementation



Implementation

- This section of the presentation will illustrate some of the original code and how it looked after the porting, concentrating on what I found particularly inter
- Due to the length and complexity of the program only selected parts are presented here



Implementation Initialisation

- The first section of the code defined study-specific macro variables, cleaned output folders, then copied files from other locations

```
%let udrug=d123;  
%let uprot=d1234r00006;  
%let prot= d1234r00006;  
title;  
  
X "rm /cars/cdm/data/&udrug/&uprot/jreview/rawdata/*.*";  
X "rm /cars/cdm/data/&udrug/&uprot/jreview/*.*";  
  
**** copy needed files from prod rawdata directory ****;  
X "cp /cars/prod/&udrug/&uprot/csr/rawdata/addct.sas7bdat  
/cars/cdm/data/&udrug/&uprot/jreview/rawdata/.";  
[...]  
X "cp /cars/prod/&udrug/&uprot/csr/rawdata/d1234r00006.sas7bdat  
/cars/cdm/data/&udrug/&uprot/jreview/rawdata/.";  
[...]  
X "cp /cars/prod/&udrug/&uprot/csr/rawdata/pkdur.sas7bdat  
/cars/cdm/data/&udrug/&uprot/jreview/rawdata/.";
```

- There are two main issues here in my view:
 - The X function is extensively used when not necessary, making the code platform-specific
 - Study code has been hardcoded when unnecessary



Implementation Initialisation (2)

- The modified code looks like this

```
%setup;
```

```
* Source of data under 'ar' - Modify as needed ;  
%let reporting_event = interim_exp1;
```

```
%let eice_path = &studydir/&delivery_namex/&execution_areax;
```

```
options sasautos=("&eice_path/macro", sasautos);
```

```
%let udrug=%scan(&_exec_programname, 2, _);  
%let uprot=%scan(&_exec_programname, 3, _.);  
%let prot=&study_namex;  
title;
```

```
**** copy needed files from prod rawdata directory ****;  
libname in_1 "%sysfunc(tranwrd(&studydir/, /dm/,  
/ar))/&reporting_event/raw/prod/data/data_area" access=readonly inencoding=any;  
libname out "&eice_path/data/rawdata";
```

```
proc datasets lib = in_1 nolist;  
  copy out=out mt=data;  
  select addct [...] &study_namex [...] pkdur;
```



Implementation Input Data Collection

- The next part of the code collects input data

```
**** copy all raw datasets from cdm study directories ****;  
%macro fcopy (folder, condition);  
  
*** read the folder with the latest raw data ****;  
filename pipedir pipe "ls /cars/cdm/data/&udrug/&uprot/&folder " 1rec1=32767;  
[...]  
***** copy data from IN to OUT *****;  
X "cp /cars/cdm/data/&udrug/&uprot/&folder/&curdt/*.*  
/cars/cdm/data/&udrug/&uprot/jreview/rawdata/.";  
%mend fcopy;  
  
%fcopy (data, %str(substr(folder,1,4)='data'));  
[...]  
%fcopy (data1, %str(substr(folder,1,5)='data1' and  
index(folder,'rejected')=0));  
  
%macro xpttosas;  
[...]  
%mend xpttosas;  
%xpttosas;  
  
X "mv /cars/cdm/data/&udrug/&uprot/jreview/rawdata/pk.sas7bdat  
/cars/cdm/data/&udrug/&uprot/jreview/rawdata/pkdur.sas7bdat";  
X "mv /cars/cdm/data/&udrug/&uprot/jreview/rawdata/im.sas7bdat  
/cars/cdm/data/&udrug/&uprot/jreview/rawdata/extimdur.sas7bdat";
```

- The X function is used extensively, and data set names, as well as their length, are hardcoded. A few referenced datasets did not actually exist



Implementation Input Data Collection (2)

- This is the revised code, based on a set of macros automatically scanning input folders for several file types and copying them

```
* Copy necessary T-domain XLS files to local folder ;  
options msglevel=i;
```

```
%macro cp_xls(_domain=);  
filename src "%sysfunc(tranwrd(&studydir/, /dm/,  
/ar))/&reporting_event/raw/prod/data/data_area/&_domain..xls" recfm=n;  
%let fref=src;  
%if %sysfunc(fexist(&fref)) %then %do;  
filename dest "&eice_path/data/rawdata/&_domain..xls" recfm=n;
```

```
* Copy the records of SRC to DEST. ;
```

```
[...]
```

```
%end;
```

```
%mend;
```

```
%cp_xls(_domain=ta);
```

```
[...]
```

```
%cp_xls(_domain=tv);
```

```
%get_everything(input_data = &studydir/raw/&execution_areax/data,  
output_path = &eice_path/data/rawdata);
```



Implementation SDTM Data Sets Creation

- The next section run the SDTM programs developed by Statistical Programming *in situ*, redirecting the output to a local folder

```
***** run SDTM programs *****;

filename sdtm "/cars/prod/&udrug/&uprot/csr/datasets/sdtm";
%include sdtm(autoexec);

** create new folder with current date as a name and set permissions to it **;
%let cdate=%sysfunc(compress(%sysfunc(date()), yymmdd10.), '-');
X "mkdir &root/cdm/data/&udrug/&uprot/jreview/sdtm/&cdate";
X "chmod 775 &root/cdm/data/&udrug/&uprot/jreview/sdtm/&cdate";

/***** re-assign autoexec.sas libraries *****/

libname in "&root/cdm/data/&udrug/&uprot/jreview/rawdata" inencoding=asciiany;
libname an "&root/cdm/data/&udrug/&uprot/jreview/rawdata" inencoding=asciiany;
libname sdtm "&root/cdm/data/&udrug/&uprot/jreview/sdtm/&cdate";

%include sdtm(dm);
[...]
%include sdtm(vs);
```

- This code has been essentially left alone, apart from removing the X statements and applying the necessary adjustments linked to the different folder structures, e.g.:

```
filename sdtm "%sysfunc(tranwrd(&studydir/, /dm/,
/ar))/&reporting_event/sdtm/&execution_areax/program";
```



Implementation Post-processing Phase

- Here the variability among the studies kicked in fully, since the programmer had to look at which data sets were present

```
***** SDTM post-processing phase for JReview *****;
```

```
***** assign jreview libraries *****;
```

```
%let cdate=%sysfunc(compress(%sysfunc(date()), yymmdd10.), '-');
```

```
X "mkdir &root/cdm/data/&udrug/&uprot/jreview/golive/&cdate";
```

```
X "chmod 775 &root/cdm/data/&udrug/&uprot/jreview/golive/&cdate";
```

```
libname in "/cars/cdm/data/&udrug/&uprot/jreview/sdtm/&cdate" access=readonly;
```

```
libname out "/cars/cdm/data/&udrug/&uprot/jreview/golive/&cdate";
```

```
libname jreview "/cars/cdm/data/&udrug/&uprot/jreview";
```

```
filename job "/cars/cdm/programming/studies/jreview/progs";
```

```
*****;
```

```
*** copy all domains from IN to OUT, check for missing labels and merge all  
parent and supp domains ***;
```

```
*****;
```

```
%include job(supp_faxx);
```

```
*****;
```

```
***** derive baselines and endpoints (BLFL & ENDFL) *****;
```

```
*****;
```

```
%include job(get_endpoints);
```

```
%get_endpoints(LB);
```

```
%get_endpoints(VS);
```

```
%*get_endpoints(EG);
```

```
[...]
```



Implementation Post-processing Phase (2)

- As mentioned earlier, there was no time to apply structural changes to the program, so only minor changes were applied.
- E.g., sections dealing with specific domains could be turned into macros, to be executed only if a domain existed



Implementation Output File Creation and Export

- The final section went through the steps necessary to create and export a zip archive to a predefined location

```
*** zip files and unzip them to CDPreports jreview upload folder ***;  
%include job(uploadzip);
```

```
* Following steps are added to transfer data to JReview upgrade cloud version ;  
libname jrsource '/SASDATAAC/cars/cdm/data/d123/d1234r00006/jreview';  
filename tranfile  
'/SASDATAAC/cars/cdm/data/d123/d1234r00006/jreview/uplift/d1234r00006.xpt';
```

```
/* CPORT files to XPT */  
proc cport library=jrsource file=tranfile memtype=data;  
run;
```

```
/* ZIP the XPT File */  
x zip -r -j /cars/cdm/data/d123/d1234r00006/jreview/uplift/d1234r00006-cdb.zip  
/cars/cdm/data/d123/d1234r00006/jreview/uplift/d1234r00006.xpt;
```

```
/* Copy ZIP XPT file to SFTP folder */  
X "cp /cars/cdm/data/d123/d1234r00006/jreview/uplift/d1234r00006-cdb.zip  
/cars/Jreview_CARS/Open/Inbound/.";
```

- Note the use of the wrong extension for a CPORT file (necessary due to several variable names being too long), as well as many X statements and way too much hardcoding



Implementation Output File Creation and Export (2)

- The ported code is as follows, with the creation of the zip archive now performed within SAS using ODS PACKAGE (comments have not been modified)

```
*** zip files and unzip them to CDPreports jreview upload folder ***;  
%include job(uploadzip);  
  
* Following steps are added to transfer data to JReview upgrade cloud version ;  
libname jrsource "&eice_path/data/data_area";  
filename tranfile "&eice_path/data/uplift/&study_namex..xpt";  
  
* COPY files to XPT ;  
proc cport library = jrsource file = tranfile memtype=data;  
run;  
  
ods package(ProdOutput) open nopf;  
ods package(ProdOutput) add file="&eice_path/data/uplift/&study_namex..xpt";  
ods package(ProdOutput) publish archive  
    properties(archive_name="%upcase(&study_namex)-CDB_JREVIEWEXP.zip"  
        archive_path="&studydir/raw/doc/export");  
ods package(ProdOutput) close;
```



5

Conclusions



Conclusions

- Since there were no functional specifications, I had to deduce the intended behaviour looking at the code rather than referencing them, which led to misunderstandings
- The original program used a number of different folders to store data sets, several of them probably not needed
- The wide use of hardcoding and Operating System-specific commands were all alleviated using macro variables and SAS functions, respectively.
- The time available to complete the porting was quite short
 - Only the minimum set of changes allowing the program to run correctly in the new system were applied
- Additional enhancements were applied for each further study I was asked to work on, but keeping track of them was not easy
- Creating formal functional specifications would have made sense if the program had been planned for a longer period of use



Questions?



Confidentiality Notice

This file is private and may contain confidential and proprietary information. If you have received this file in error, please notify us and remove it from your system and note that you must not copy, distribute or take any action in reliance on it. Any unauthorized use or disclosure of the contents of this file is not permitted and may be unlawful. AstraZeneca PLC, 1 Francis Crick Avenue, Cambridge Biomedical Campus, Cambridge, CB2 0AA, UK, T: +44(0)203 749 5000, www.astrazeneca.com

