**PhUSE 2010**

**Paper PO07**

# Data Migration from Oracle Clinical to .XML using SAS

Matteo Ferrari, CROS NT, Verona, Italy
Valeria Visonà, CROS NT, Verona, Italy

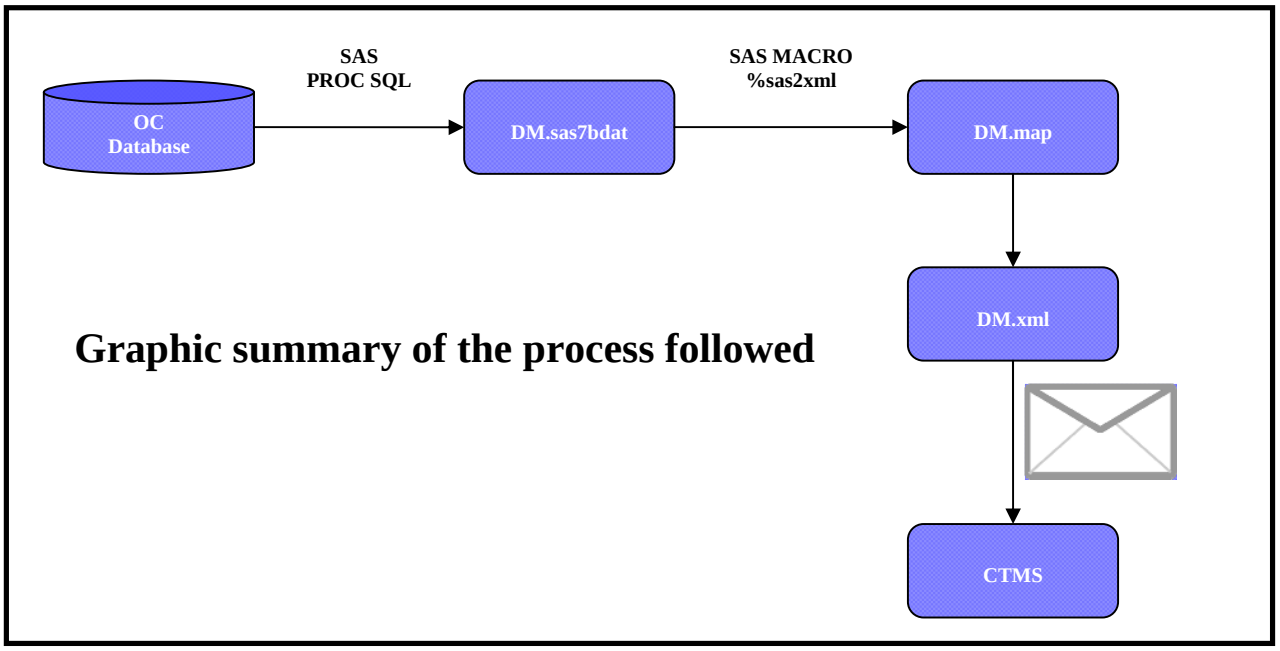
## ABSTRACT

The main strength of XML data structure consists in allowing the user greater flexibility in managing data in several platforms. The work performed and presented in this poster was motivated by a Sponsor's request to export data contained in an Oracle Clinical® database into an .XML file. The Sponsor needed to own a data structure that could be used in different systems without changing the raw source data. We successfully managed to satisfy this request using SAS® as a connection between the two types of data.
The aim of this poster is to present the processes followed, showing the SAS programs through which the final goal was attained. Furthermore, we would like to present the SAS program developed which allows you to automatically send the .XML file previously created, through the company mail server to the final user.

## INTRODUCTION

Every platform for managing data used in the daily work has its own peculiarity. For a more flexible way to work it is useful to convert the data into a general data structure that can be easily used with several systems. In our experience to satisfy the requirements of the client, XML (eXtensible Markup Language) data structure enabled us to reach our goal.

This standard language was needed to be used by the study monitors in the Clinical Trial Management System (CTMS) to maintain and manage the planning, preparation, performance, and reporting of the clinical trial; in this way monitors can easily use the data for different aims, such as scheduling of visits, assessments, etc.
In the flux diagram presented below, the process followed starting from OC raw data to the delivery of the final product is summarized.



## FIRST STEP: FROM ORACLE TO SAS

The first step consists in importing data into SAS format starting from Oracle Clinical data structure. This operation can be achieved using the SAS PROC SQL syntax to create a connection between the OC and the SAS structure.
In the following, we restrict ourselves to presenting this process limited to certain information collected in the clinical database. In particular, in the next rows of SAS code, some variables regarding the subjects' demography (sex, age, date of birth) are loaded from Oracle and saved into DM.sas7bdat dataset.

```
proc sql;
    connect to oracle(path='instance' user='username' pw='password');
    create table DATA.dm as select
    STUDY     as STUDY      label ="Clinical Study"     format  $15.
    ,INVSITE  as INVSITE    label ="Site"               format  $10.
    ,PT       as PT         label ="Patient"            format  $10.
    ,VISIT    as VISIT      label ="Visit"              format  10.
    ,AGEN     as AGEN       label ="Age"                format 3.
    ,PTINIT   as PTINIT     label ="Patient Initials"   format $3.
    ,SEX      as SEX        label ="Sex"                format $2.
    ,BRTHDTC  as BRTHDTC    label ="Date of Birth"      format $200.

    from connection to oracle (select
    STUDY STUDY,
    INVSITE INVSITE,
    PT PT,
    VISIT_NUMBER VISIT,
    AGEN AGEN,
    PTINIT PTINIT,
    BRTHDTC_FUL BRTHDTC,
    SEX SEX
    from STUDY$CURRENT.DM);
    disconnect from oracle;
quit;
```

**Definition of attributes of variables that will be collected in DM.sas7bdat**

**Definition of the correspondence of OC variables and the new created SAS ones.**

The resulting SAS dataset is shown below. It contains all the variables specified in the PROC SQL, formatted as defined by the user.

| | Clinical Study | Site | Patient | Visit | Age | Patient Initials | Sex | Date of Birth |
|---|---|---|---|---|---|---|---|---|
| 1 | PROT001 | SITE001 | 1 | 1 | 28 | ABC | F | 19810711 |
| 2 | PROT001 | SITE001 | 2 | 1 | 28 | ABC | M | 19810208 |
| 3 | PROT001 | SITE001 | 3 | 1 | 37 | ABC | M | 19720929 |
| 4 | PROT001 | SITE001 | 4 | 1 | 56 | ABC | F | 19530602 |
| 5 | PROT001 | SITE001 | 5 | 1 | 46 | ABC | F | 19630523 |
| 6 | PROT001 | SITE001 | 6 | 1 | 66 | ABC | F | 19430731 |
| 7 | PROT001 | SITE001 | 7 | 1 | 66 | ABC | M | 19431024 |
| 8 | PROT001 | SITE001 | 8 | 1 | 54 | ABC | F | 19550324 |
| 9 | PROT001 | SITE001 | 9 | 1 | 66 | ABC | F | 19430206 |
| 10 | PROT001 | SITE001 | 10 | 1 | 58 | ABC | F | 19510822 |

## SECOND STEP: FROM SAS TO XML

The second step of the process focuses on converting the SAS dataset DM into an .XML file; to do this, the macro *%sas2xml* was developed. This macro needs three input parameters:

- ds → name of the SAS dataset to be converted;
- tags → specify the names of the tags separated by a '/'. The first one indicates the root tag that corresponds to the biggest level, in this case the dataset. The second one indicates the child tag that corresponds to a level contained in the previous one. This tag indicates what the rows of the dataset represents, in this case the patient;
- path → path in which the output files will be saved.

In the following, the *%sas2xml* SAS macro is presented:

```
1  %macro SAS2XML(ds=,tags=,path=);
2  /* Starting from DM.sas7bdat, a dataset containing variables' attributes (name, type, length) is
3     created through PROC CONTENTS */
4
5  proc sort data=contents; by VARNUM; run;
6
7  data _null_;
8     set contents end=eof;
9     by VARNUM ;
10
11    colName=trim('<COLUMN '||compress(trim('name="'||NAME||'">'),' '));
12    PathName=trim('<PATH syntax="XPath">'||compress(trim('/'||tagsname||'/'||NAME||'</PATH>'),' '));
13    tabPath =trim('<TABLE-PATH syntax="XPath">'||compress(trim('/'||tagsname||'</TABLE-PATH>'),' '));
14
15    file "&path.\&ds..map";
16
17    * Creation of the header of the .MAP file --> see PART 1;
18    if _n_ = 1 then do;
19       put '<?xml version="1.0" encoding="windows-1252"?>';
20       put '<SXLEMAP name="' "&ds." '" version="1.2">' ;
21       put '<TABLE name="' MEMNAME'">';
22       put tabPath;
23    end ;
24
25    * Creation of the body of the .MAP file --> see PART 2;
26    if <condition> then do;
27    .
28    .
29    .
30    end;
31
32    * Creation of the footer of the .MAP file --> see PART 3;
33     if eof then do;
34    .
35    .
36    .
37    end;
38 run;
39
40 FILENAME out "&path.\&ds..xml";
41 FILENAME map "&path.\&ds..map";
42 LIBNAME out XML92  XMLTYPE=XMLMAP XMLMAP=MAP;
43
44 data out.&ds.;
45  set data.&ds.;
46 run;
47 %mend SAS2XML;
48 %SAS2XML(ds=DM,tags=DEMOG/PATIENT,path=\\Path_of_output_files);
```

**Creation of the .MAP file starting from a dataset containing the list of variables and their attributes.**

**Creation of the .XML file by means of the .MAP file**

The code presented above (lines 2-38), let the user build a support file (DM.map). This file is created to maintain the structure of the input dataset (DM.sas7bdat) in the .XML output.
A block of this .MAP file is presented below. Between <COLUMN> and </COLUMN>, each variable of the input dataset is described and its own characteristics defined.

```
<?xml version="1.0" encoding="windows-1252"?>
<SXLEMAP name="DM" version="1.2">
<TABLE name="DM ">
<TABLE-PATH syntax="XPath">/DEMOG/PATIENT</TABLE-PATH>

<COLUMN name="STUDY">
<PATH syntax="XPath">/DEMOG/PATIENT/STUDY</PATH>
<TYPE>character</TYPE>
<DATATYPE>string</DATATYPE>
<LENGTH>13 </LENGTH>
</COLUMN>

.
.
.
.
.
.

</TABLE>
</SXLEMAP>
```

PART 1: HEADER

**Characteristics of the first variable collected in the DM dataset.**

PART 2: BODY

PART 3: FOOTER

Through the statements from line 40 to 46 of the *%sas2xml* macro, the .XML file is created.
Firstly the references to the .XML and .MAP files are created (i.e. *filename* statement).
In a second step with the *Libname* statement a SAS reference is associated to a .MAP file in order to successfully export the .XML document. The *XMLMap* syntax tells the XML engine how to transform SAS dataset, variables and observations into XML markup.

A block of this .XML file is presented below.
Between *<DEMOG>* and *</DEMOG>* (root tag) all the observations derived from the SAS dataset are included.
Whereas between *<PATIENT>* and *</PATIENT>* (child tag), all the information regarding a single observation of the source dataset is reported.

```
<?xml version="1.0" encoding="windows-1252" ?>

<!--
      SAS XML Libname Engine (SAS92XML)
      SAS XMLMap Generated Output
  -->

<DEMOG>
   <PATIENT>
      <STUDY>PROT001</STUDY>
      <INVSITE>SITE001</INVSITE>
      <PT>1</PT>
      <VISIT>1</VISIT>
      <AGEN>28</AGEN>
      <PTINIT>ABC</PTINIT>
      <SEX>F</SEX>
      <BRTHDTC>19810711</BRTHDTC>
   </PATIENT>

   <PATIENT>
      <STUDY>PROT001</STUDY>
      <INVSITE>SITE001</INVSITE>
      <PT>2</PT>
      <VISIT>1</VISIT>
      <AGEN>28</AGEN>
      <PTINIT>ABC</PTINIT>
      <SEX>M</SEX>
      <BRTHDTC>19810208</BRTHDTC>
   </PATIENT>

   .
   .
   .
   .


</DEMOG>
```

**Variables' values for the first observation collected in the DM dataset.**

### THIRD STEP: AUTOMATIC SENDING OF THE .XML FILE THROUGH SAS

Finally the .XML file is ready for the delivery.
In order to save time, avoiding repetitive operations such as the preparation of the e-mail for the sponsor, including attachment of the file, specification of the addresses and the text, further SAS steps were implemented.

```
1  filename outbox email ;
2
3  data _null_ ;
4     file outbox
5     from="matteo.ferrari@cros.it"
6     to=("valeria.visona@cros.it")
7     cc=("info@cros.it")
8
9     subject="XML File Delivery"
10    attach="\\Path_of_output_files\DM.xml" ;
11
12    put 'Dear Valeria,';
13    put 'please find attached the XML file containing demographic data of PROTO01 study.';
14    put 'Kind regards,';
15    put 'Matteo';
16
17 run;
```

## SCHEDULING OF SAS PROCEDURES

The Sponsor specifically request the production and the delivery of the .XML file daily. To achieve this, the daily scheduling of the programs presented in this poster on a SAS server was implemented; saving time and avoiding the daily manually running of the programs.

## CONCLUSION

A sponsor's request sometimes can represent a starting point for the development of new processes. In this case we created a macro system that allowed us to export data and send by e-mail to the sponsor only running a SAS program. Furthermore, by scheduling the program the user is not required to run the program themselves!

## CONTACT INFORMATION

Your comments and questions are valued and encouraged.

Contact the authors at:
Matteo Ferrari & Valeria Visonà
CROS NT S.r.l.
Via Germania 2, Verona 37136
Office Phone: 0039 0458202666 Fax: 0039 0458205875
Emails: matteo.ferrari@cros.it valeria.visona@cros.it Web: www.cros.it