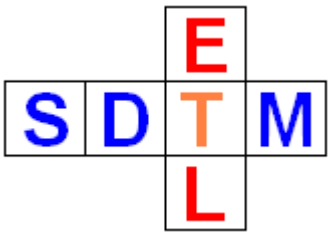
SDTM-ETL 5.2 User Manual and Tutorial
'SDTM-ETL Light' and running in batch execution mode



Author: Jozef Aerts, XML4Pharma

Last update: 2026-05-27

Table of contents:

Introduction

This document describes how to run the transformations for generating SDTM from the command line using the CLI (Command Line Interface) or in batch, and how to use the "SDTM-ETL Light" version of the software, for executing mappings and generate SDTM files, but without working on the mappings themselves.

Running SDTM-ETL execution in batch mode / CLI mode

With SDTM-ETL, it is also possible to execute the mappings, and generate SDTM datasets from existing mappings, in line command or batch mode, e.g. on a server, using commands or scripts that are executed using the CLI (Command Line Interface).

Although it is easily possible to execute the software for generating SDTM/SEND datasets by typing the command from the command line, one will usually want to store the commands in a file (".bat" file on Windows, ".sh" file on Linux/Unix) and run that file. Such files are usually designated as "batch file".

The contents of such a batch file for will be very similar to the SDTM-ETL.bat file that is used to start SDTM-ETL in GUI mode, but there are also a good number of differences, as the execution command has a good number of additional parameters.

First of all, the batch file should set the libraries for executing the software, for example:

The base for executing in batch mode is given by:

```
java -Xms256M -Xmx1024M -cp SDTM-ETL.jar com.xml4pharma.sdtmetl.light.SDTMETLExecutor ...
```

When however running from another directory then the one where the software is located, one needs to provide the full path to the "SDTM-ETL.jar" file, e.g.

```
java -Xms256M -Xmx1024M -cp D:\SDTM-ETL_5_0\SDTM-ETL.jar
com.xml4pharma.sdtmetl.light.SDTMETLExecutor ...
```

Remark that this is slightly different from the base in earlier versions!

The -Xms256M and -Xmx1024M parameters indicate the amount of memory the java VM is allowed to claim, the first being the amount at start, and the latter being the maximum allowed at all. 1024MB will be sufficient in most cases, but the user may decide to either a lower value or increase the amount of memory claimed by the Java Virtual Machine (VM), depending on the amount of mappings and the sizes of the files to be transformed.
It is advised never to allow more than about 50% of the available physical memory, as the operating system usually also uses a lot of memory.

The command is followed by the set of keywords and parameters. For example, in the case SAS-XPT must be generated:

```
java -Xms256M -Xmx1024M -cp  SDTM-ETL.jar
com.xml4pharma.sdtmetl.light.SDTMETLExecutor
-DEFINEVERSION 2.0
-GENERATECOMMENTSDOMAIN
-DEFINEFILELOCATION C:\SDTM-ETL\TestFiles\MyStudy_Six_Domains_define_2_0.xml
-CLINICALDATAFILELOCATION C:\SDTM-ETL\TestFiles\ODM1-3\MyStudy_ODM_1_3.xml
-STUDYID MyStudy
-ODMVERSION 1.3
-METADATAFILELOCATION C:\SDTM-ETL\TestFiles\ODM1-3\MyStudy_ODM_1_3.xml
-GENERATESASXPT
-SASXPTDIRECTORYLOCATION C:\SASXPTSamples\
-ADAPTLENGTHFORLONGESTVALUE
-GENERATERELRECDOMAIN
-GENERATECOMMENTSDOMAIN
-GENERATESUPPQUALDOMAINS
```

All these should be in a single line (no carriage return).

Or when Dataset-JSON 1.1 must be generated:

```
java -Xms256M -Xmx1024M -cp  SDTM-ETL.jar
com.xml4pharma.sdtmetl.light.SDTMETLExecutor
-DEFINEVERSION 2.0 -GENERATECOMMENTSDOMAIN
-DEFINEFILELOCATION C:\SDTM-ETL\TestFiles\MyStudy_Six_Domains_define_2_0.xml
-CLINICALDATAFILELOCATION C:\SDTM-ETL\TestFiles\ODM1-3\MyStudy_ODM_1_3.xml
-STUDYID MyStudy -ODMVERSION 1.3
-METADATAFILELOCATION C:\SDTM-ETL\TestFiles\ODM1-3\MyStudyNew_ODM_1_3.xml
-GENERATEDATASETJSON
-DATASETJSONDIRECTORYLOCATION C:\temp\
-GENERATERELRECDOMAIN
-GENERATECOMMENTSDOMAIN
-GENERATESUPPQUALDOMAINS
```

I.e. each keyword starts with a dash ("-") and is, depending on the keyword itself, followed by a parameter value.
The list of keywords and parameters is given in the following table. Those given in bold-italic are ***mandatory***. If one of these is absent, the program will issue a warning and stop.

All parameters/keywords may be given either in lower case or mixed case or uppercase.
They will however be translated to uppercase before interpretation by the program.

Parameter / keyword	**Parameter value**	**Explanation**

Parameter / keyword	Parameter value	Explanation
-DEFINEFILELOCATION	file path	absolute or relative path of the define.xml file containing the mappings. See also remark on file paths.
-CLINICALDATAFILELOCATION	file path	absolute or relative path of the file containing the clinical data. See also remark on file paths.
-METADATAFILELOCATION	file path	absolute or relative path of the file containing the metadata (i.e. study design). This can be identical to the path for the clinical data when the file contains both metadata and clinical data.
-ADMINDATAFILELOCATION	file path	only necessary when the mappings use data from the administrative data section of the study, and this section is not included in the file with clinical data. The former is essentially the case when the SDTM-ETL functions "sitename()" is used in one or more mappings. See also remark on file paths.
-XMLOUTPUTFILELOCATION	file path	absolute or relative path of the file where the SDTM/SEND data in XML format will be written to. Required when the output is in Dataset-XML format. See also remark on file paths.
-STUDYID	OID of the study	The OID of the study, as given in the Study/@OID attribute of the file with metadata (or ClinicalData/@StudyOID in the file with clinical data).
-ODMVERSION	version of the ODM standard used for the study metadata and clinical data	Default is "1.2". Can be either "1.2", "1.3" or "1.3.1". Support for ODMv2 will be added in the near future.
-DEFINEVERSION	Version of the define.xml file containing the mappings	Can be "1.0" or "2.0" or "2.1" Remark that Define-XML v.1.0 is outdated and has very limited support in SDTM-ETL 5.0.

Parameter / keyword	Parameter value	Explanation
-GENERATESUPPQUALDOMAINS	none	keyword to indicate that supplemental qualifier datasets should be generated for: a) automated splitting of records with more than 200 characters (usually necessary when also SAS XPT datasets need to be generated) b) non-standard SDTM variables (NSVs) that need to be moved to the corresponding SUPP-- domain. Remark that when this keyword is omitted, no splitting is done, which may be problematic when output is in SAS XPT format.
-GENERATERELRECDOMAIN	none	keyword to indicate that a RELREC domain needs to be automatically created for "RELREC" variables (see user manual for further details).
-GENERATE_1_N_RELREC	none	keyword to indicate that the software should attempt to generate 1:n relationships in the automatically generated RELREC domain. Only to be used in combination with the -GENERATERELRECDOMAIN keyword.
-GENERATECOMMENTSDOMAIN	none	keyword to indicate that a Comments (CO) domain needs to be automatically generated from "Comment" variables (see user manual for further details). Remark: As of SDTM-ETL v.4.4, this keyword must be set explicitely. Until SDTM-ETL v.3.3, when generating SAS XPT datasets, CO datasets were always generated automatically.
-GENERATESASXPT	none	keyword to indicate that SAS XPT SDTM/SEND datasets should be generated. It is recommended to then also use the keyword "-GENERATESUPPQUALDOMAINS" as otherwise records of more than 200 characters will be cut after the 200th character.
-SASXPTDIRECTORYLOCATION	file path	absolute or relative path of the directory where the SAS XPT files need to be written to. Only to be used in combination with the "-GENERATESASXPT" keyword
-GENERATEDATASETJSON	none	keyword to indicate that datasets must be generated in the modern CDISC Dataset-JSON-1.1 format
-DATASETJSONDIRECTORYLOCATION	file path	absolute or relative path of the directory where the Dataset-JSON files need to be written to. Only to be used in combination with the "-GENERATEDATASETJSON" keyword

Parameter / keyword	Parameter value	Explanation
-GENERATEUTF8CSV	none	keyword to indicate that UTF-8 encoded CSV should be generated. Either "-GENERATESASXPT" or "-GENERATEDATASETXML" or "-GENERATEUTF8CSV" must be provided
-UTF8CSVDIRECTORYLOCATION	file path	absolute or relative path of the directory where the UTF-8 encoded files need to be written to. Only to be used in combination with the "-GENERATEUTF8CSV " keyword
-UTF8CSVADDHEADERLINES	none	add 2 header lines to the UTF-8 encoded CSV file, one with the variable names, and one with the variable labels
-GENERATESQLINSERT	none	keyword to indicate that "SQL Insert" statements need to be created and stored in a file or folder
-GENERATESINGLESQLFILE	none	keyword that all "SQL Insert" commands must be written into a single file, even when this is for different datasets.
-SQLINSERTDIRECTORYLOCATION	file path	absolute or relative path of the directory where the file(s) with "SQL Insert" statements need to be written to. Only to be used in combination with the --GENERATESQLINSERT " keyword. In case also -GENERATESINGLESQLFILE is provided, the software expects that the value is a path to a file. In case a folder is provided anyway, the single SQL file will be named based on the study-ID (e.g. "CES_INSERT.sql". If -GENERATESINGLESQLFILE is not provided, the software expects that a directory (folder) is provided.
-USEORACLESQL	none	For generating "SQL Insert" statements, use Oracle SQL. The default is to generate "standard" SQL that is compatible with MySQL.
-KEEPSTUDYIDPREFIXFORSQLTABLE-NAME (as one word …)	none	for the table name, the default is that the study-id is removed. For example, if the study ID is "CES", and thus the study-specific data definition OIDs are e.g. "CES:DM", "CES:VS", then the table names will be "DM" and "VS". When however the keyword is added, the table names will be "CESDM" and "CESVS" respectively, I.e. the colon is removed from the OID to form the table name.

Parameter / keyword	Parameter value	Explanation
-REPLACECOLONFORTABLENAMEBY	one or more characters (no blanks)	ONLY to be used in combination with "-KEEPSTUDYIDPREFIXFORSQLTABLENAME". The provided parameter value will then replace the colon from the OID to form the table name. For example, when one provides: "-REPLACECOLONFORTABLENAMEBY _" (i.e. the "underscore" character), the table names will e.g. be "CES_DM" and "CES_LB" respectively.
-REMOVEPREFIXFROMCOLUMNNAME	none	In SDTM-ETL, SDTM/SEND variables are defined by their OID, that have a "domain prefix", for example "VS.VSTESTCD". This will also be the default column name in the generated SQL table. If one wants to only have the variable name as the column name, for example "VSTESTCD", one must add the parameter -REMOVEPREFIXFROMCOLUMNNAME
-REPLACEDOTINCOLUMNNAMEBY	one or more characters (no blanks)	One can also have the dot "." in the column name replaced by one or more characters (without blanks). For example, when using: "-REPLACEDOTINCOLUMNNAMEBY _", the table name in the SQL Insert statements will e.g. be "VS_VSTESTCD". Do not use in combination with -REMOVEPREFIXFROMCOLUMNNAME
-USETYPEDITEMDATA	none	keyword to indicate that the clinical data use "typed ItemData" elements (e.g. ItemDataInteger" in the ODM file with clinical data. May only be used when ODMVERSION is either 1.3 or 1.3.1 (so not for ODM version 1.2). Do NOT use for ODMv2.
-ISOPENCLINICAVERSION	none	keyword to indicate that support needs to be provided for the OpenClinica 3.x extensions to the ODM standard. Can only be used with the "OpenClinica version" of the software.
-POSTPROCESSINGFORLOBXFL	none	keyword to indicate that --LOBXFL values must be generated using postprocessing for Findings datasets. Requires that the corresponding --LOBXFL variable is define in the define.xml that drives the dataset generation.

Parameter / keyword	Parameter value	Explanation
-USELOINCFORLOBXFLPOSTPROCESS-ING (as one word …)	none	keyword to indicate that the LOINC code (and if absent, only LBTESTCD) should be used to define "test uniqueness" for the calculation of --LOBXFL. Default is that the combination of --TESTCD, --CAT, --SCAT, --METHOD, --SPEC, --LOC, --LAT), --DIR, --POS and --OBJ, is used for "test uniqueness" (FDA rule FDAB026).
-GENERATEMERGEDDATASETSSPLITDOMAINS (as one word …)	none	keyword to indicate that in case that there are "split domain" datasets to be generated (e.g. LBUR, LBCH, LBHE), also a "merged" dataset (and possible the corresponding merged SUPP-- dataset) needs to be generated. This dataset than gets the name of the domain, extended by "_merg" (e.g. LB_merg.xpt) **ONLY use when you have "split domain" datasets.**
-RESTARTSEQSPLITDOMAINDATASETS	none	keyword to indicate that in the case of "split domain" datasets, the --SEQ values for each subject must restart at "1" in each dataset. Remark that the default is to have unique sequence number across different datasets for the same domain. So, having this keyword corresponds to having the checkbox "Unique --SEQ values across 'split domains' unchecked. This keyword will e.g. be used in the case that one has different QSxx datasets, for which the general practice is to treat them as non-related to each other, so that one does not want to have unique sequence numbers, but one wants to have --SEQ (re)starting from 1 for each subject in each of the QSxx datasets.
-ADAPTLENGTHFORLONGESTVALUE	none	Only for when SAS XPT or Dataset-JSON files are generated. In the case of XPT: takes care that the XPT files are optimized for the actual lengths of the variable values. In the case of Dataset-JSON: takes care that "longest value length" is used for adding to the metadata section of the Dataset-JSON file. Corresponds to the checkbox "Adapt variable length for longest result value" in the GUI version. Do **NOT** use for UTF-encoded CSV or SQL generation.

Parameter / keyword	Parameter value	Explanation
-RESORTUSINGDEFINEXMLKEYS	none	When added, the system will try to sort the generated datasets according to the "keys" (as provided by the KeySequence attribute on the "ItemRef" element. Currently, this has only been implemented for SAS-XPT. Use with care! The choice of the "keys" needs to be made very carefully. Inappropriate choice of the keys may lead to unexpected results or the software may crash.
-GENERATEUNSCHEDULEDVISITNUMS	none	When added, the system will try to automatically generate fractional VISITNUM values for unscheduled visits. These need to be assigned an empty value in the mapping script for VISITNUM, e.g. $VS.VISITNUM=''; and the assignment "UNSCHEDULED" in the mapping script for VISIT. E.g. $VS.VISIT = 'UNSCHEDULED'; Usually, this will require that the --DTC is a "key" in the define.xml, and that sorting is done using the keys, by also setting the "-RESORTUSINGDEFINEXMLKEYS" parameter.
-USEZEROZEROONEINCREMENT-FORUNSCHEDULEDVISITNUM (as one word …)	none	For unscheduled visits, the default increment for the VISITNUM value is 0.1. This will assign subsequent unscheduled visits the VISITNUM values of e.g. "5.1", "5.2" etc.. When this parameter is added, the increment value is set to 0.01, which will lead to VISITNUM values of e.g. "5.01", "5.02", ...
-ADDVISITNUMUNSCHEDULEDTOVIS-ITNAME (as one word …)	none	When automatically assigning VISITNUM values for unscheduled visits, the value for "VISIT" normally remains "UNSCHEDULED" as present in the mapping script. In case this parameter is set, the word "UNSCHEDULED" will be extended with the generated VISITNUM value, so the value for VISIT will then e.g. become "UNSCHEDULED 5.1"

Parameter / keyword	Parameter value	Explanation
-VISITNAMEFROMVISITNUMSEPARA-TIONCHARACTER (as one word …)	text	when using the parameter "-ADDVISITNUMUNSCHEDULEDTOVISITNAME" (as one word) the default character between "UNSCHEDULED" and the fractional visitnum value will be a blank. When added, this parameter with an additional parameter value (text) will replace the blank by the text provided in the parameter value. When e.g. the parameter value is "-", the result will be: UNSCHEDULED-5.2. Remark that for batch execution, the text string should not have blanks in it. This is only possible when using the GUI.
-BASEVISITNUMUNSCHEDULED	integer	base visit number (integer) for the case of unscheduled visits before the first scheduled visit. Default value: 0. For example, if one chooses the value -2 for the base visit number, then, with the increment for unscheduled visits being 0.1, then the earliest unscheduled visit before the first scheduled visit will get VISITNUM=-1.9 (-2 + 0.1)
-USEFIRSTCOMPLIANTSVVISITNUM-FORUNSCHEDULED (as one word …)	none	This parameter will be used when there is an SV dataset definition present, that comes before any other dataset definitions for which unscheduled VISITNUM values need to be generated automatically, and there are 'overlapping' visits, which often occur when the date part for one of the dates (--DTC in the dataset or --STDTC in SV) is missing. In such a case, the unscheduled VISITNUM will be based on the last SV visit for which the --DTC of the non-SV visit is in the window (SVSTDTC-SVENDTC) of the SV visit.
-USELASTCOMPLIANTSVVISITNUM-FORUNSCHEDULED (as one word …)		This parameter will be used when there is an SV dataset definition present, that comes before any other dataset definitions for which unscheduled VISITNUM values need to be generated automatically, and there are 'overlapping' visits, which often occur when the date part for one of the dates (--DTC in the dataset or --STDTC in SV) is missing. In such a case, the unscheduled VISITNUM will be based on the first SV visit for which the --DTC of the non-SV visit is in the window (SVSTDTC-SVENDTC) of the SV visit. This is the default.

Running "batch" execution from within a directory other than the SDTM-ETL installation directory

Until before v.5.0, batch execution had to be started from within the directory where the SDTM-ETL software is installed. On request of a number of users, we have changed the mechanism to start batch execution so that it can be started from any directory, or using a ".bat" file in any directory.

To achieve this, ensure that the path to the SDTM-ETL.jar in the command is an absolute path, e.g.

java -Xms256M -Xmx1024M ***-cp D:\SDTM-ETL_5_0\SDTM-ETL.jar*** com.xml4pharma.sdtmetl.light.SDTMETLExecutor ...

Secondly, make a copy of the file "properties.dat" in the directory where you start the batch execution from. We have chosen for this to allow users to have different setting (as defined in the properties.dat file) for different use cases. For example, one could have different folders with ".bat" files, for each study one, and have an adapted "properties.dat" file for each individual study.

Make sure that the directory where you start the batch execution from has write access: in most cases, the software will create a "temp" directory in that directory to store intermediate files, as well as a "logs" directory with the log files.

Remark that there is no need to make a copy of the license file (license.dat) in the directory where batch execution is started from: the license file must only reside in the directory of the software installation

Using "batch" execution in production

When having a process in place for doing the batch execution in production mode, it may be wise to use a define.xml file with the template removed, i.e. only keeping the study-specific domain/dataset definitions, this as loading the template takes considerable time.
One can generate such a "template-free" define.xml in SDTM-ETL using the menu "File - Save define.xml for batch execution".

Of course, one should always test the batch execution carefully before taking it into production.

Example batch files

Example batch files can be found in the distribution. These can be used as start for your own files, but need to be adapted for your own use case, especially for the file paths. Following files are available:

Batch_example_SDTM-ETL_SASXPT.bat	Example file for generation of SDTM files in SAS XPT files
Batch_example_SDTM-ETL_DatasetJSON.bat	Example file for generation of SDTM files in modern Dataset-JSON format.
Batch_example_SDTM-ETL_DatasetXML.bat	Example file for generation of SDTM files in modern Dataset-XML format
Batch_example_SDTM-ETL_SASXPT_define_2-1.bat	Example file for generation of SDTM files in SAS XPT files from a Define-XML 2.1 file with mappings

Batch_example_SDTM-ETL_SASXPT_RWS.bat	Test example file using mappings that use RESTful web services for generating SDTM records

These example batch files use example ODM and Define-XML files from the "TestFiles" folder that is included in the distribution.

Remark concerning file paths

File paths can be absolute or relative with respect to the directory from which the batch execution is run.
In case the file path contains one or more blanks, the whole path should be embedded in double quotes. For example:

... -DEFINEFILELOCATION "C:\Documents and Settings\My Define Files\MyStudyDefine.xml"

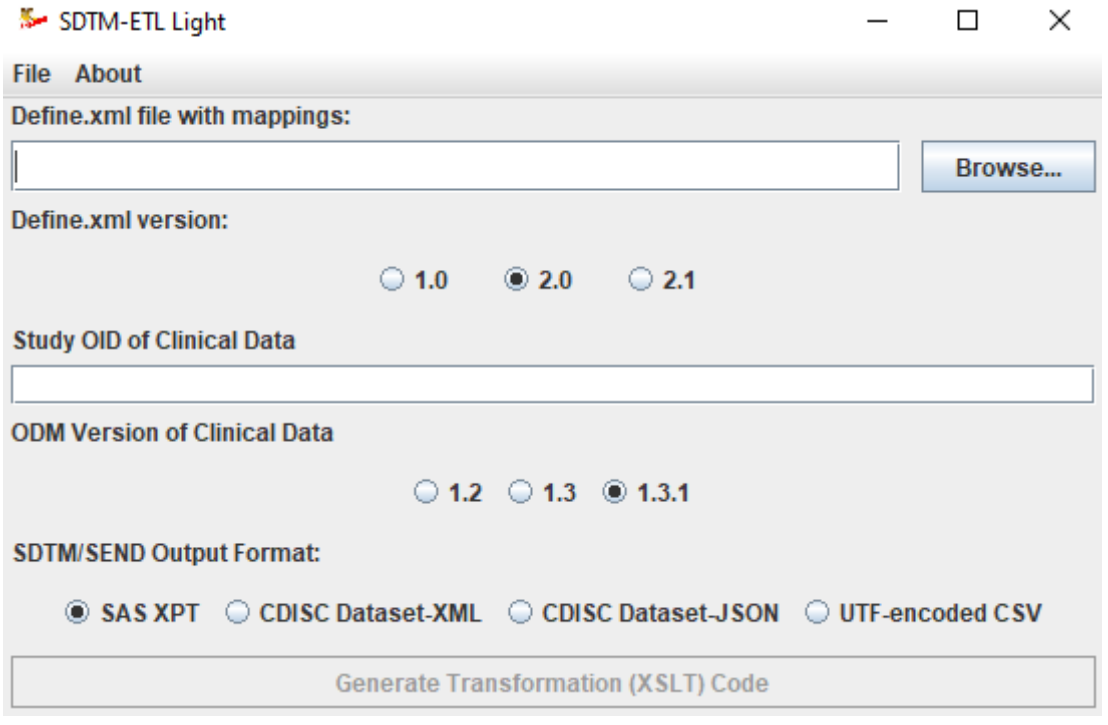
Current limitations

Batch execution currently only has been tested extensively for SAS-XPT output.
For Dataset-JSON 1.1, UTF-8-CSV, and SQL-Insert, the implementation is still experimental.

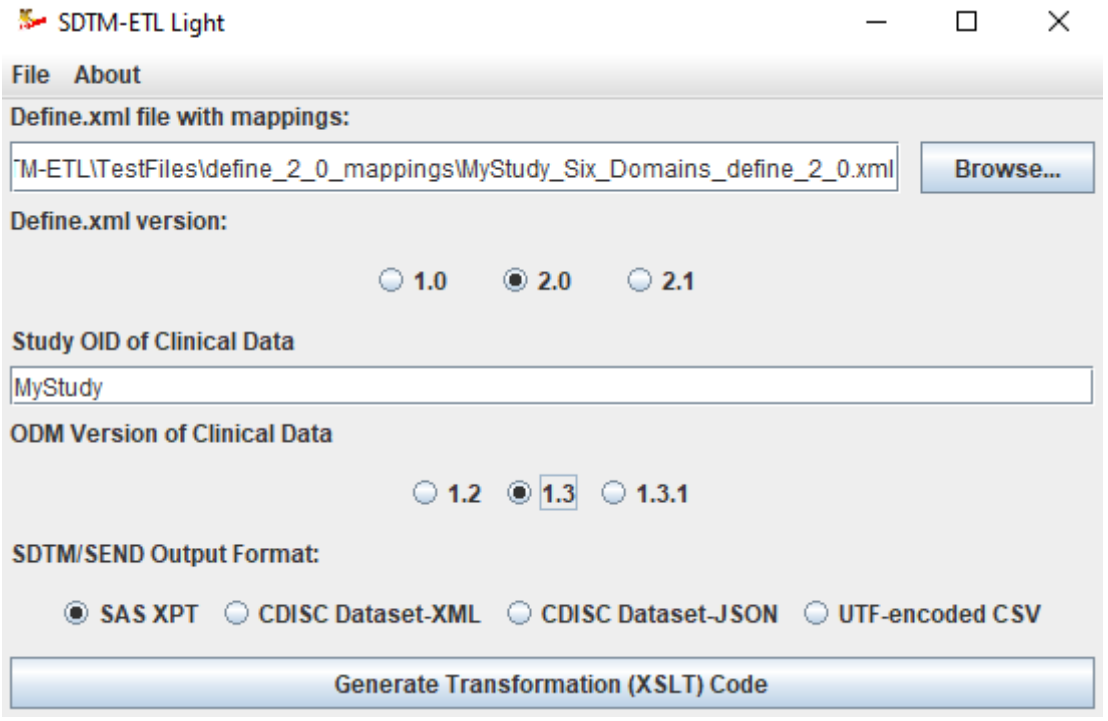
Starting transformation-execution-only from a simple graphical user interface ("SDTM-ETL Light")

Whereas the instructions provided above are ideal for executing the transformations on a server, e.g. at regular times using a cron job scheduler, some people may also want to execute the mappings on a normal PC or workstation, but without working on the mappings themselves, i.e. just executing them. For this use case, we developed a simple graphical interface. This "light" version can be started using the file "***SDTM-ETL_execution_only_GUI.bat***".

When started, the following graphical user interface is shown:

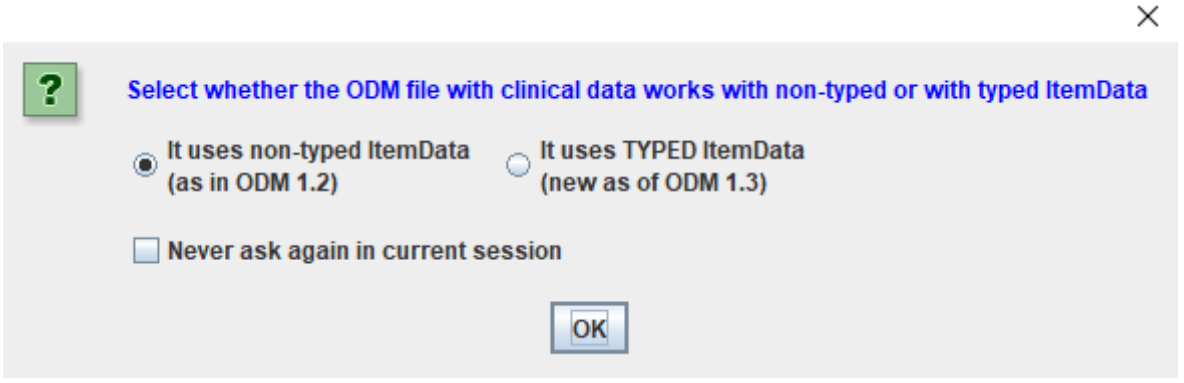


One should then fill in all fields and select the appropriate checkboxes, e.g.:

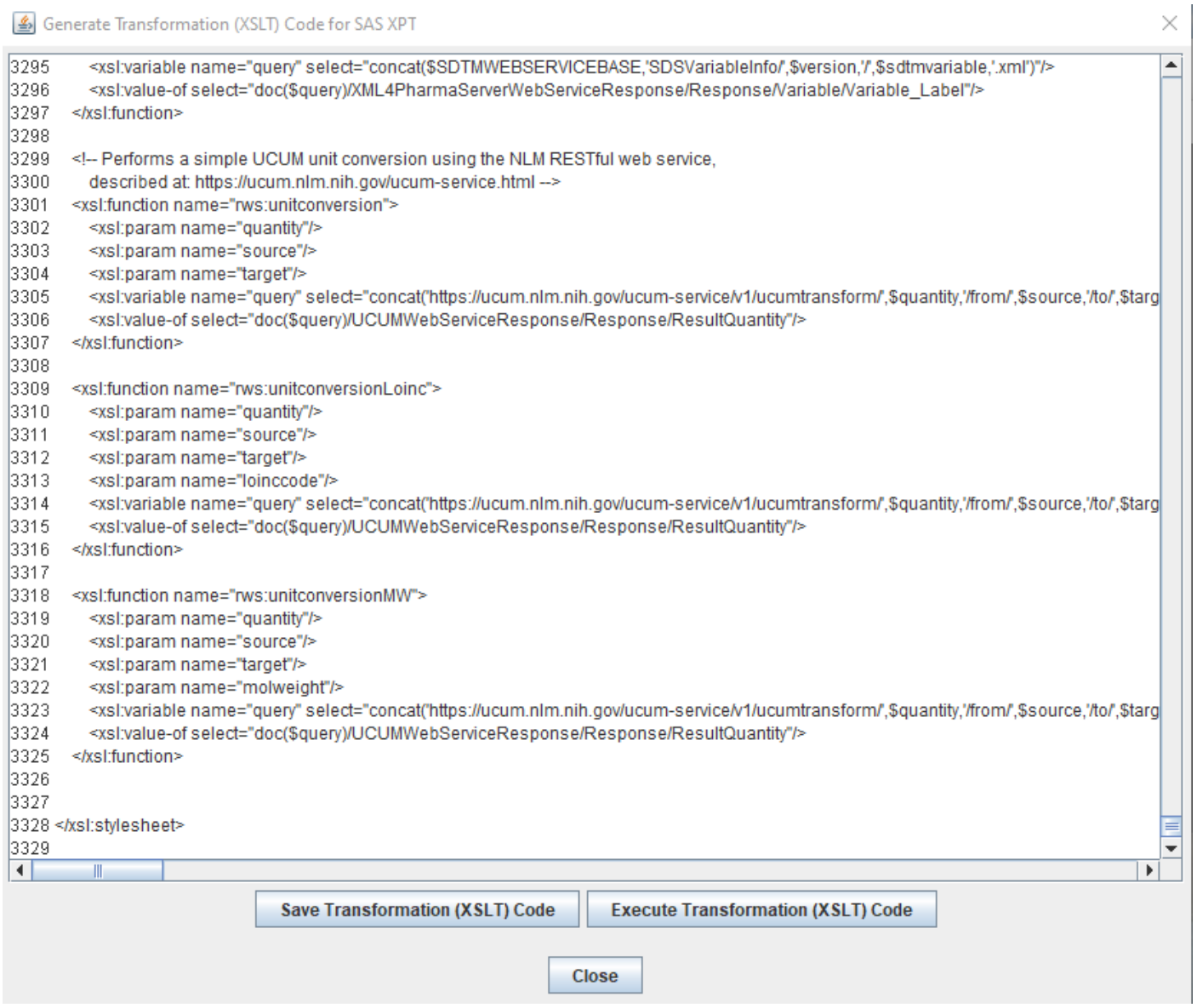


And can then choose between "SAS XPT" (SAS Transport 5), CDISC's own Dataset-XML or Dataset-JSON format, and UTF-8 encoded CSV for the output format.

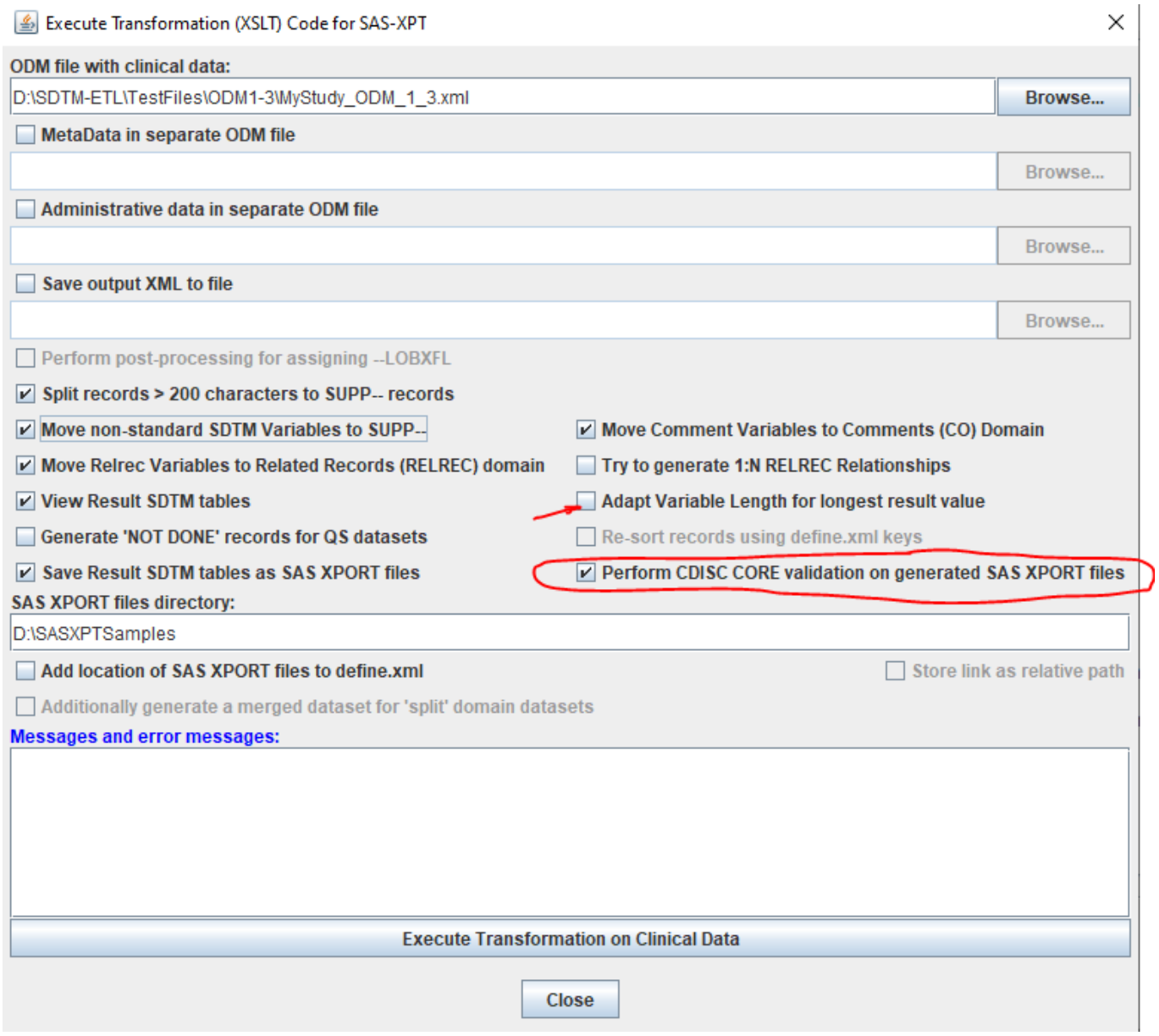
When the clicking the "Generate Transformation (XSLT) Code, the dialogs that we already know from the full version pop up:



followed by:



Upon clicking the "Execute Transformation (XSLT) Code" button, the classic dialog for entering source and target and the different options is displayed:



As usual, one can select several "special" options, such as optimizing the XPT files for file size[1], using the checkbox "Adapt Variable Length for longest value", generation of "NOT DONE" records in the case of QS datasets, and generation of "1:N" RELREC relationships".

New in SDTM-ETL 4.2 is that one can ask the system to re-sort the records according to the "keys" ("KeySequence" attributes) in the define.xml, which should only be necessary when the data in the source ODM is not in chronological order). The checkbox for this option will only be made visible when the system detects such keys in the define.xml file from which the mapping execution is started.
Another new feature in SDTM-ETL is to perform CDISC CORE validation on the generated datasets, essentially replacing the "crappy" Pinnacle21 validation engine.
See the separate documents "SDTM-ETL 4.2 new features overview" and "Validating SDTM/SEND datasets using CDISC-CORE" on the SDTM-ETL website.

The final step is then to click the button "Execute Transformation on Clinical Data" which starts the transformation execution.

For further details about the last steps, see the tutorials on the SDTM-ETL website.

---

[1] This is due to that the outdated SAS-XPT is a highly inefficient format, where values, shorter than the variable length, are padded with blanks until the given length is reached.

CLI and Batch / SDTM-ETL Light execution of SDTM-ETL v.5.0