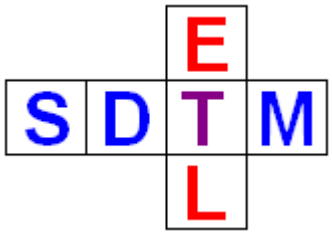
SDTM-ETL™



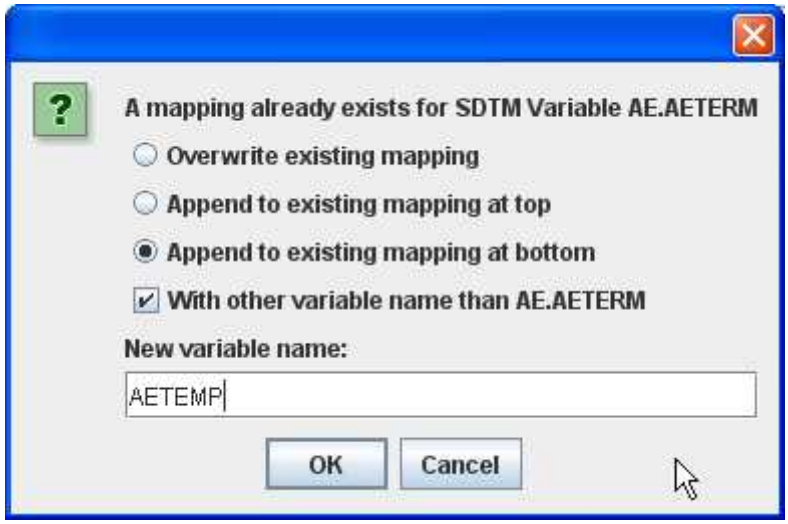
New features in version 1.4

Author: Jozef Aerts – XML4Pharma
May 2010

Table of Contents

Improvements during drag-and-drop

When doing a drag-and-drop from the ODM tree to an SDTM variable (cell in the table) for which a mapping already exists, the user is now given the opportunity to append the new mapping to the existing mapping (instead of inserting it at the top), and with another variable name than the standard SDTM name.



This makes it easier (and faster) to combine data from the ODM tree to generate mappings for SDTM variables, as it reduces the amount of manual editing of the generated mapping script.

System remembers "Generalize for all ..."

In case the information for an SDTM variable needs to be retrieved from the same ODM "ItemData" but e.g. for different visits ("StudyEvents"), the menu "Generalize for ..." is usually used. When doing a second drag-and-drop into the same domain, the system now remembers which "Generalize for ..." checkboxes were checked in the previous drag-and-drop.
This allows faster working and also reduces the probability that an error is made during the definition of a mapping.

Alias visible on ItemDef in ODM tree

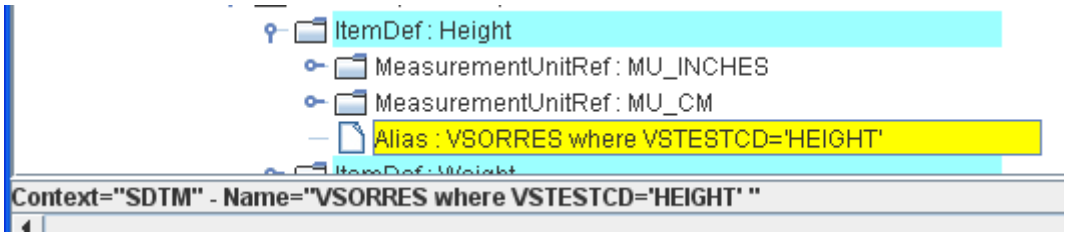
The usage of the "Alias" element becomes more and more important. It allows to define a "synonym" in a certain "context" for the item.
The CDISC ODM team has recommended to use the "Alias" to annotate items with as well CDASH, SDTM as information for the use in interaction with electronic health records.
In our case, the usage for SDTM annotations is especially important, as in mappings where a 1:1 mapping is not possible, the information in "SDSVarName" is insufficient[1].
For example:

1 As SDSVarName is limited to 8 characters and must be a valid SAS name.

```
- <ItemDef DataType="integer" Length="3" Name="Height" OID="I_HEIGHT" SASFieldName="HEIGHT"
    SDSVarName="VSORRES">
  + <Question>
  + <MeasurementUnitRef MeasurementUnitOID="MU_INCHES">
  + <MeasurementUnitRef MeasurementUnitOID="MU_CM">
  + <RangeCheck Comparator="LT" SoftHard="Hard">
  + <RangeCheck Comparator="LT" SoftHard="Hard">
    <!--  Alias as SDTM annotation   -->
    <Alias Context="SDTM" Name="VSORRES where VSTESTCD='HEIGHT'" />
```

When one has an Item "Height" which will need to be mapped to VSORRES, the information in "SDSVarName" is insufficient, as it does not say anything about for which VSTESTCD the item will need to be used.
Therefore, one can annotate the Item with the additional information "VSORRES where VSTESTCD='HEIGH'" which is a more complete designation of what the item will be needed for in the context of SDTM. So an "Alias" is used, with the attribute "Context" having the value of "SDTM" and the annotation itself given in the "Name" attribute[2].

In the ODM tree this is visualized as:



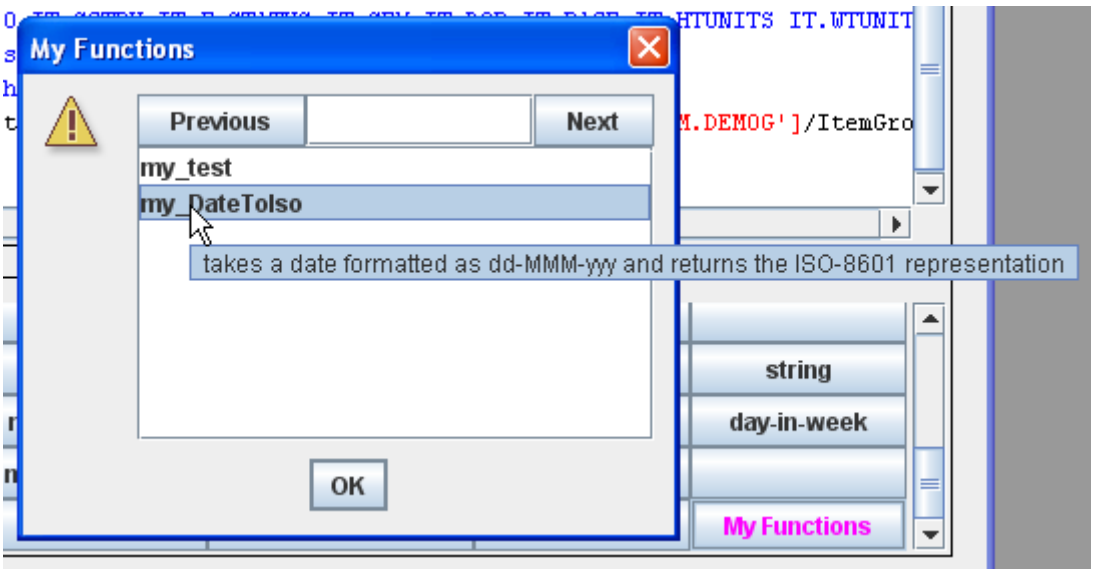
Easier access to "user" functions

Version 1.3 already allowed the definition of user functions for the mapping script and add these to the file "functions.xsl". As a conventions, such user functions should start with "my_", for example a user function "my_DateToIso" which allows the user to define a function to transform his own formatted dates into ISO-8601 presentation.

Although extremely useful, and used my many of our users, it required that the user remembers the name of the function when working with the scripting editor.

Therefore, in the scripting editor, in the panel where the functions are displayed, an extra button has been added "My Functions", which when clicked, pops up a new window containing all defined user functions (read from the "functions.xsl" file).

2 See also further in the section on finding "hot candidates"



The upper field in the new window is an "intellisense-like" search field allowing to quickly find a certain function. When a function selected, a tooltip is displayed. The text in the tooltip is taken from the first comment line ("<!-- -->") in the function definition as found in the "functions.xsl" file.

The function can then be selected and added to the script in exactly the same way as any other build-in functions.

Extended HTML view of the underlying define.xml

In version 1.3, the HTML view of the underlying define.xml was limited to what is also available in the stylesheet that was publised by CDISC. We found however that this amount of information is usually insufficient.
For example, "Length" and "SignificantDigits" and "Role" are not displayed by default, although this can be important information.
Therefore we added the possibility of an extended view of the underlying define.xml in HTML format, which also contains this non-default information.
This extended view can be obtained by checking the "Extended view" checkbox when using the menu "View -> Define.xml as HTML".

HTML View of define.xml can now be saved to file

When generating an HTML view of the underlying define.xml structure, the generated HTML can now be saved to file for offline inspection, using the button “Save HTML”.

New mapping script functions “sitename()”

A new build-in function “sitename()” has been added which takes the OID of the site (e.g. from the ODM SubjectData/SiteRef/@LocationOID which is covered by the site() function, or from an ODM data point) as an argument.
A lookup is then done in the “AdminData” section of the ODM file with metadata and the name of the location for the given OID is returned.

For example, if one has the AdminData section containing:

```
- <Location OID="LOC.site001" Name="Fred Hutchinson" LocationType="Site">
    <MetaDataVersionRef StudyOID="MyStudy" MetaDataVersionOID="v1.1.0" EffectiveDate="2001-10-19" />
  </Location>
- <Location OID="LOC.site002" Name="Roswell Park" LocationType="Site">
    <MetaDataVersionRef StudyOID="MyStudy" MetaDataVersionOID="v1.1.0" EffectiveDate="2001-10-19" />
  </Location>
```

and the new function “sitename()” is used for a mapping to “DM.SITEID”, then the result may look similar to:

SDTM Tables

MyStudy:DM | MyStudy:QS | MyStudy:SV | MyStudy:PE | MyStudy:AE | MyStudy:VS

SUBJID	DM.RFSTDTC	DM.RFENDTC	DM.SITEID	DM.BRTHDTC
001	2006-04-01	2006-05-12	Roswell Park	1960-04-03
002	2006-04-02	2006-05-02	Roswell Park	1947-02-14
003	2006-04-03	2006-05-03	Roswell Park	1972-12-01
004	2006-04-04	2006-05-04	Roswell Park	1948-06-19
005	2006-04-04	2006-05-05	Roswell Park	1963-08-20
006	2006-04-06	2006-05-06	Roswell Park	1955-01-01
007	2006-04-07		Fred Hutchinson	1970-08-09
008	2006-04-08	2006-05-08	Roswell Park	1937-02-09
009	2006-04-09	2006-05-09	Fred Hutchinson	1954-11-16
010	2006-04-10	2006-05-10	Fred Hutchinson	1958-09-17
011	2006-04-11	2006-05-11	Fred Hutchinson	1966-11-11
012	2006-04-12	2006-05-12	Fred Hutchinson	1949-03-24

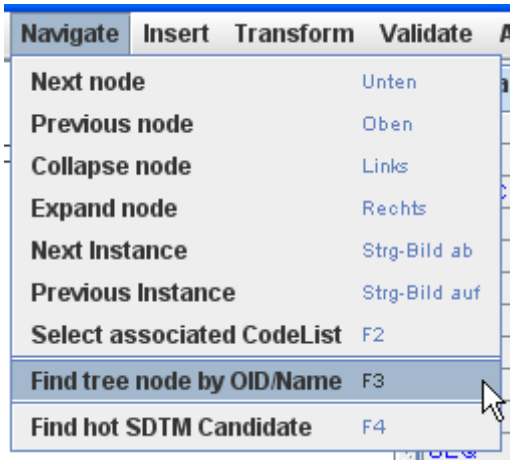
Remark that when executing the mapping, the system needs to have access to the “AdminData” of the study. If the latter is not in the same file as the metadata, do not forget to give the location of the file by checking the checkbox “Administrative Data in separate ODM file”.

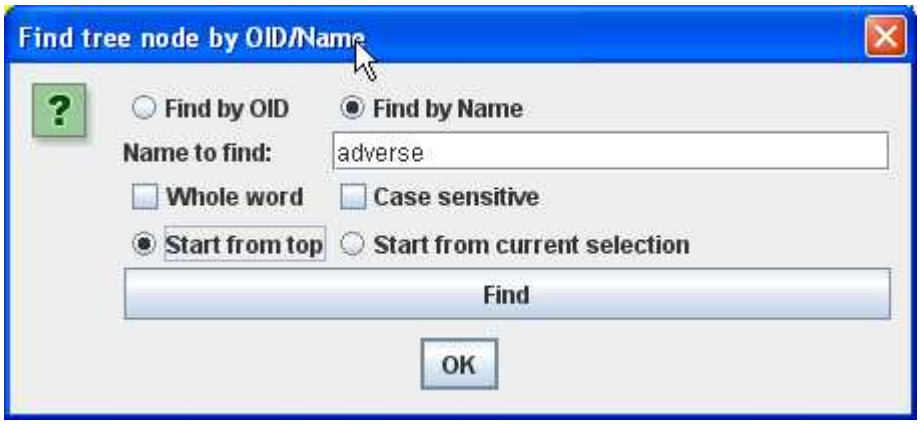
New functionality: “Navigate – Find tree node by OID/Name”

Unfortunately, some EDC vendors providing ODM export capabilities still have the tendency to assign non-meaningful OIDs to Forms, ItemGroups, Items etc..
This makes it difficult to find which item is a good candidate for a mapping to SDTM (especially if

the ODM metadata do not contain SDTM annotations[3]).

The “Name” attribute however usually contains some more useful information about what the item is about. Therefore we added a feature to search for a specific “OID” or “Name” in the ODM tree. This new functionality can be invoked by using the menu “Navigate -> Find tree node by OID/Name”, which pops up a new window:





One can either start from the top in the ODM tree, or starting from the currently selected tree node. When a hit is found, the ODM tree is expanded, and the item that is found is automatically selected. It can then be used immediately for drag-and-drop to the SDTM table.

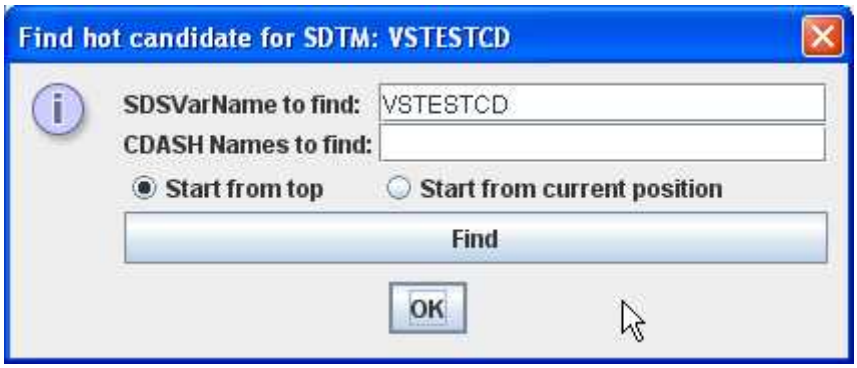
New functionality: “Navigate – Find hot SDTM candidate”

Similar to the previous functionality is the functionality “Find hot SDTM candidate”.
It allows to quickly find a suitable item in the ODM tree when the latter is annotated either using “SDSVarName”.
As CDASH is more and more implemented, we also added the functionality to search for the CDASH name, which will be found if the items are annotated[4] using:

<Alias Context=”CDASH” Name=”...”/>

3 Adding SDTM annotations to the ODM metadata is very good practice, however not very custom yet by EDC vendors.
4 This kind of annotation is highly recommended by the CDASH-ODM team

P.S. Remember that “hot candidates” (as defined by the SDSVarName attribute) are already highlighted using a square in the ODM tree.

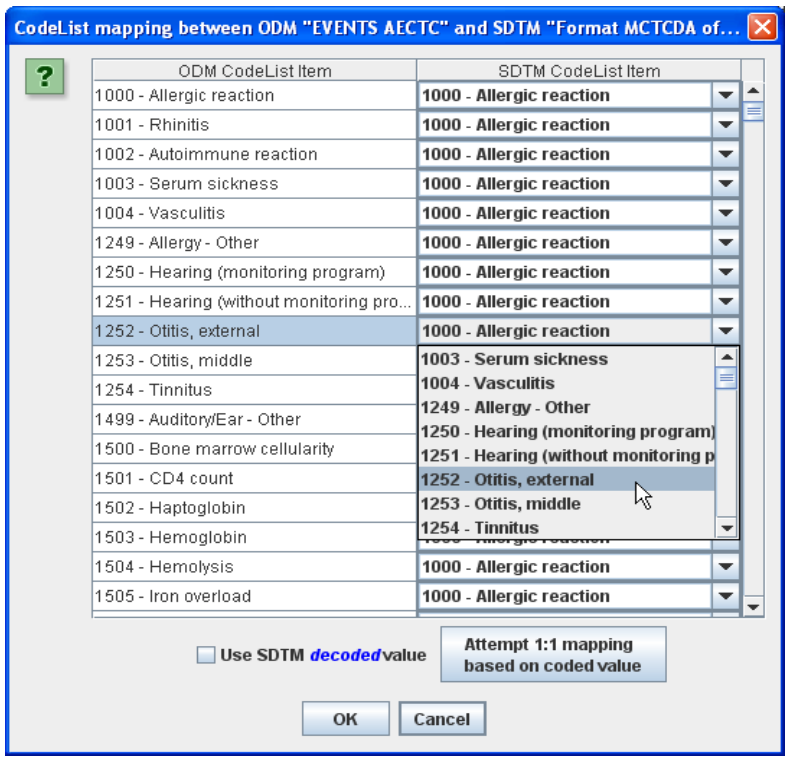


Fast codelist mapping for large codelists

In most cases, mapping between two codelists (one from the ODM and one from the SDTM) is technically pretty straightforward[5]: the codelist mapping wizard is very easy to use, and generates the mapping script automatically.

On request of a specific customer, we added functionality to allow mapping between (very) large codelists more easily.

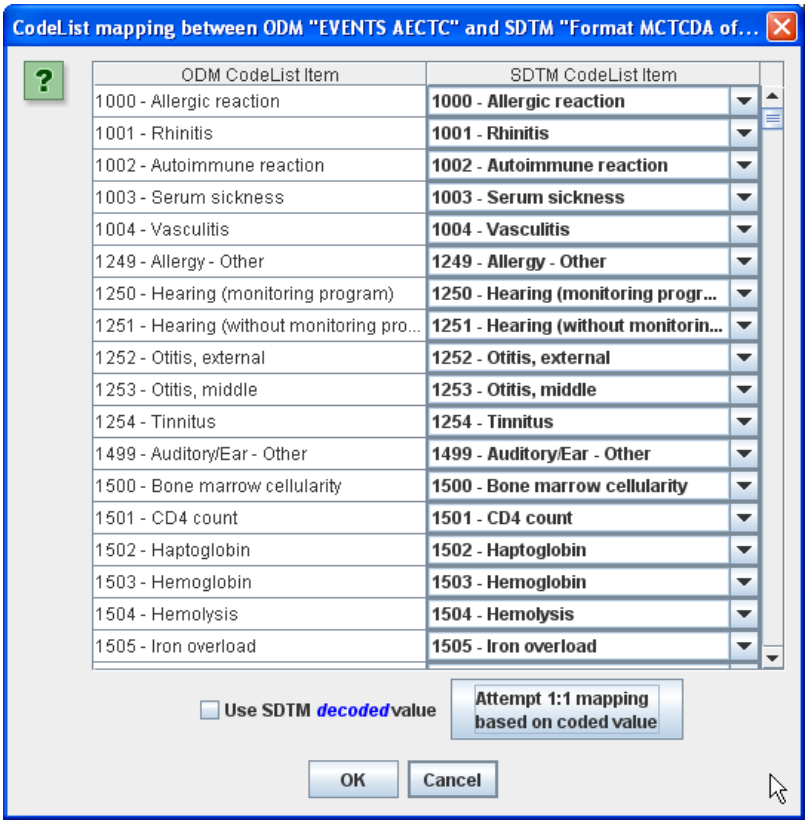
An example is shown below. In the ODM, the AE CTC codelist has been used, but not completely (only a subset was used), and also, the decoded texts are not always the same as in the published AE CTC list.



5 It can however be very difficult if the codelists in the ODM and in the SDTM are pretty or completely different, such as often is the case when (pre-SDTM) legacy data need to be mapped.

One could now start going through the whole list, and for each coded item on the left search for the corresponding one on the right, which however usually has the same coded value.
This is of course a lot of (manual) work.

In order to speed up this process, one can click the “Attempt 1:1 mapping based on coded value” button, which results in:



So the system has attempted to select the comboboxes on the right side automatically, assuming that the coded values on the left side have the same meaning as the coded values on the right side.
Of course it remains the responsibility of the user to check that this indeed the case!
Corrections can of course still be made.

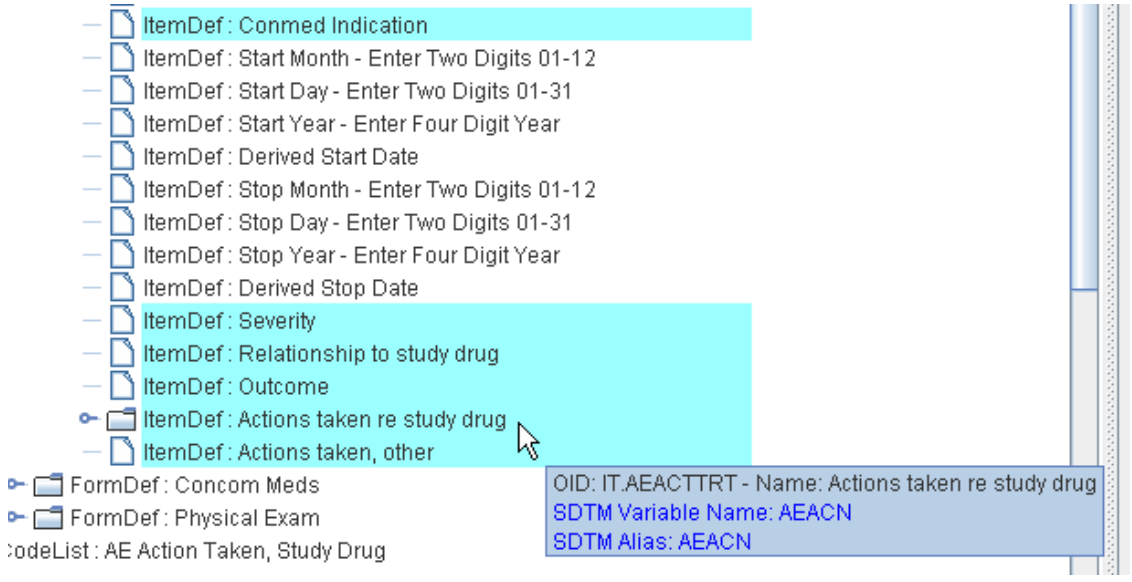
Also remark the checkbox “use SDTM decoded values”. If it is checked, a decoding to the SDTM decoded values is provided, so that the generated script will look like:

```
# Using DECODED values for the SDTM codelist$NEWCODEDVALUE = '';
if ($CODEDVALUE == '1000') {
  $NEWCODEDVALUE = 'Allergic reaction';
} elsif ($CODEDVALUE == '1001') {
  $NEWCODEDVALUE = 'Rhinitis';
} elsif ($CODEDVALUE == '1002') {
  $NEWCODEDVALUE = 'Autoimmune reaction';
} elsif ($CODEDVALUE == '1003') {
  $NEWCODEDVALUE = 'Serum sickness';
} elsif ($CODEDVALUE == '1004') {
  $NEWCODEDVALUE = 'Vasculitis';
} elsif ($CODEDVALUE == '1249') {
  $NEWCODEDVALUE = 'Allergy - Other';
} elsif ($CODEDVALUE == '1250') {
  $NEWCODEDVALUE = 'Hearing (monitoring program)';
} elsif ($CODEDVALUE == '1251') {
  $NEWCODEDVALUE = 'Hearing (without monitoring program)';
} elsif ($CODEDVALUE == '1252') {
  $NEWCODEDVALUE = 'Otitis, external';
} elsif ($CODEDVALUE == '1253') {
  $NEWCODEDVALUE = 'Otitis, middle';
} elsif ($CODEDVALUE == '1254') {
  $NEWCODEDVALUE = 'Tinnitus';
} elsif ($CODEDVALUE == '1499') {
  $NEWCODEDVALUE = 'Auditory/Ear - Other';
} elsif ($CODEDVALUE == '1500') {
  $NEWCODEDVALUE = 'Bone marrow cellularity';
```

This “decoding” is especially important when generating the mappings for the “--TEST” variables in the Findings domains from the “--TESTCD variables”.

CDASH/SDTM coloring

Items in the ODM tree that have been annotated using the “Alias” element with a “Context” value of “SDTM” or of “CDASH”, or for which the attribute “SDSVarName” is populated, are given a lightblue background color. Also a special tooltip is provided showing these annotations.
For example:

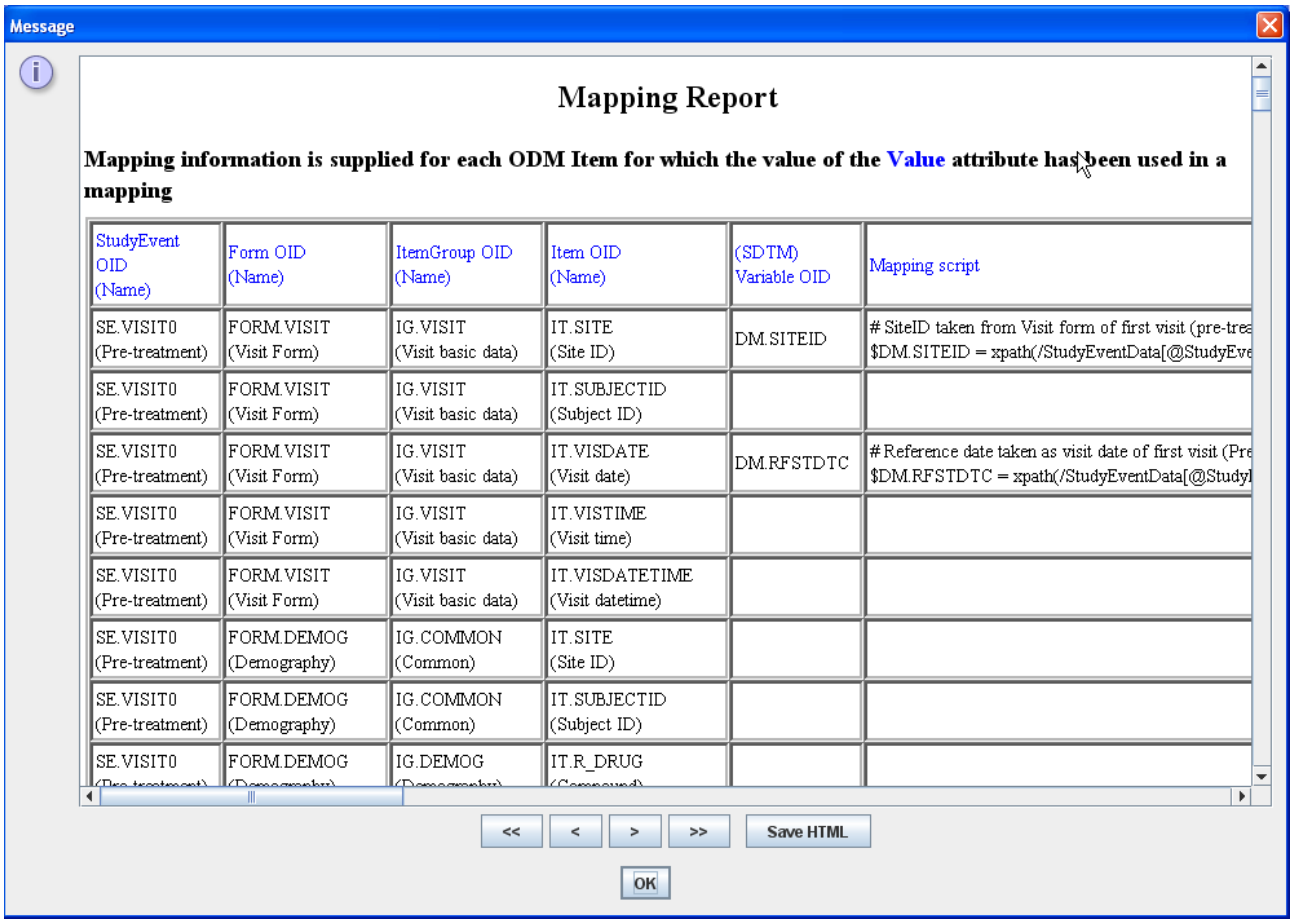


One sees in this example that the items “Conmed Indication”, “Severity”, “Relationship to study drug”, “Outcome”, “Actions taken re study drug” and “Actions taken, other” all have been annotated with either SDTM or CDASH information.
When when moves the mouse over the item “Actions taken re study drug”, the tooltip shows which annotation exactly was added.

This kind of annotations is especially important as it is a first step to come to automated generation of SDTM domains from standardized CDASH forms in future.

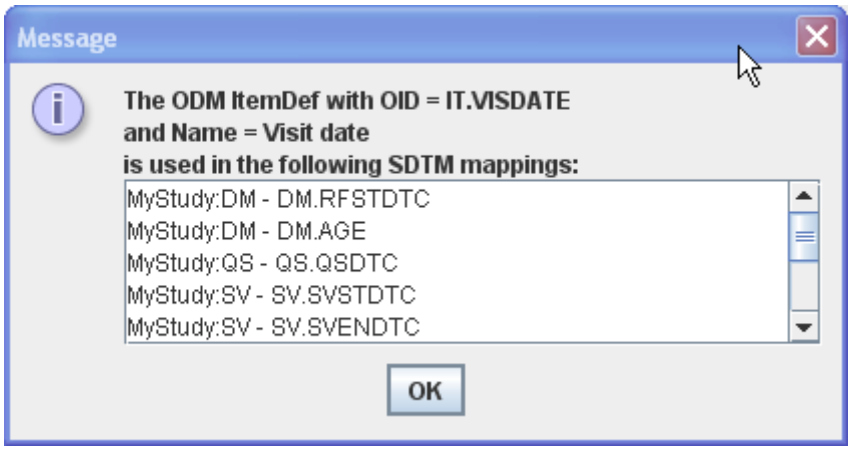
Generation of a “mapping completeness report”

Although ODM items that have been used in a mapping to an SDTM variable are “greyed out” in the ODM tree, it is often very handy to get a complete overview of which ODM items have been used in mappings (and to which SDTM variables).
Such a report (as HTML) can now be generated using the new menu “View -> Mapping Completeness Report”.
As the generation of the report can take some time, is generated in the background, and a new window pops up when the generation is complete:



For each ODM Item instance, it is shown to which SDTM variable is mapped, and the mapping script is provided.
The HTML report can of course be saved to file for offline inspection, e.g. by colleagues.

Remember that for a single Item, one can also obtain the SDTM variables to which it is mapped by doing a right-click on that Item in the ODM tree. This results in e.g.:



“Traffic lights” can be switched off

The so-called “traffic lights”, colored icons indicating whether an ODM Item is suitable (according to its datatype and associated codelist) for mapping with a selected SDTM variable can be switched off by using the menu “View -> ODM Items without 'traffic lights'”.
This can be an interesting option in case of a large ODM (study design) which makes the recalculation of the suitability (and the recoloring of the “traffic lights”) slow when another SDTM variable is selected.
After switching off the “traffic lights” the system should then become considerably faster.

The “traffic lights” can be switched on again by using the menu “View -> ODM Items with 'traffic lights'”.

SDTM and define.xml validation using OpenCDISC

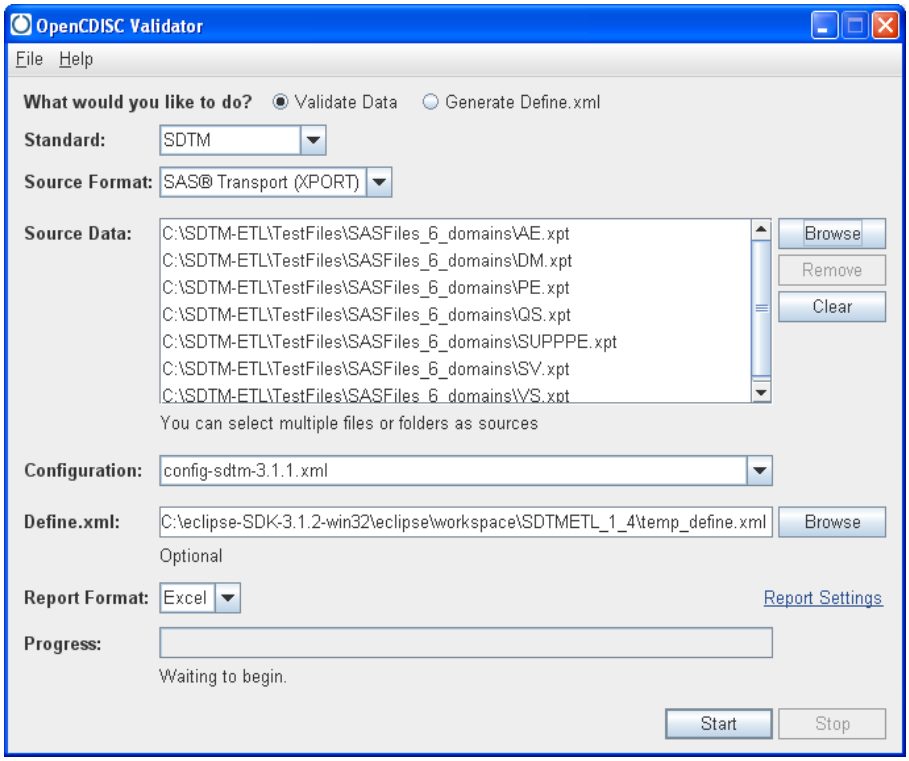
The “OpenCDISC Validator[6] v.1.0” has now been integrated into the SDTM-ETL software.

It allows as well to validate SDTM datasets (SAS XPT format) against the WebSDM rules as to validate the underlying define.xml against the define.xml standard.
Although SDTM-ETL has a build-in define.xml validator, using the OpenCDISC validator for define.xml is also interesting, as the rules of define.xml are not always 100% clear and sometimes open for interpretation. So validation using a validator that was independently developed is always a very good option[7].

The interface with the OpenCDISC Validator can however also be used for validating the generated SAS datasets against the SDTM standard, as defined by the published WebSDM rules and the SDTM-IG.

6 See www.OpenCDISC.org. With special thanks to Max Kanevsky of OpenCDISC for the permission to integrate the OpenCDISC Validator into SDTM-ETL.
7 In our opinion, the OpenCDISC Validator rules for define.xml are sometimes a bit too strict. For example, it gives a warning when “ImputationMethod” is used, as this ODM element in not mentioned in the define.xml specification. However we exactly use “ImputationMethod” to store our mappings in.

When using the menu “Validate – Validate SAS-SDTM Datasets using OpenCDISC”, the OpenCDISC Validator software is started in a separate window:

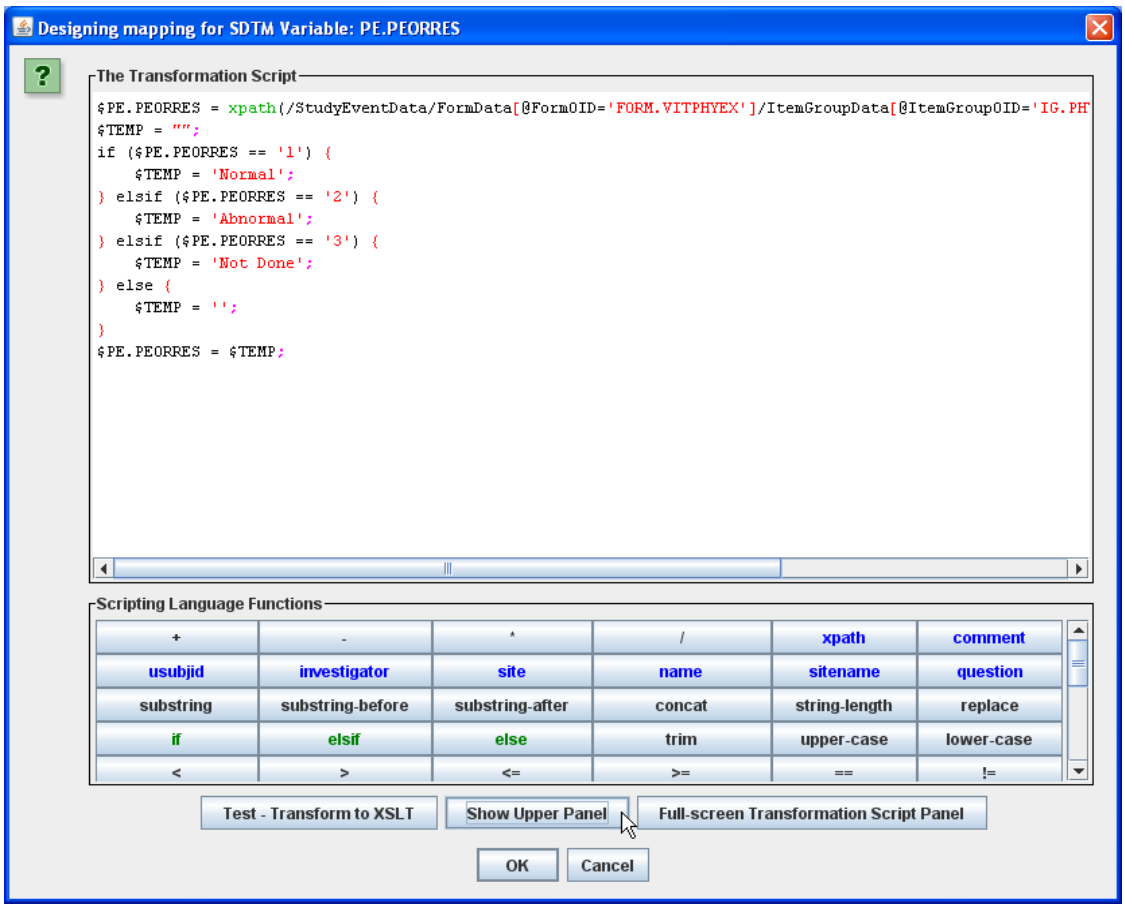


One should then populate the field “Source Data” with the SAS XPT SDTM-datasets that one want to validate. Also check that the correct “Configuration” is selected, depending on whether one uses SDTM-IG 3.1.1 or SDTM-IG 3.1.2 (SDTM 1.1 or 1.2).
The field “Define.xml” is prepopulated, as at the moment the OpenCDISC Validator is started, our underlying define.xml is written to a temporary file (only the study-specific domains) which is then used as the metadata for the SDTM datasets.
One can then still choose how reporting is done: either using an Excel output file, or as CSV, or as HTML. In our experience, the most extensive report is generated when the option “Excel” is selected.
The validation can then be started by clicking the “Start” button.

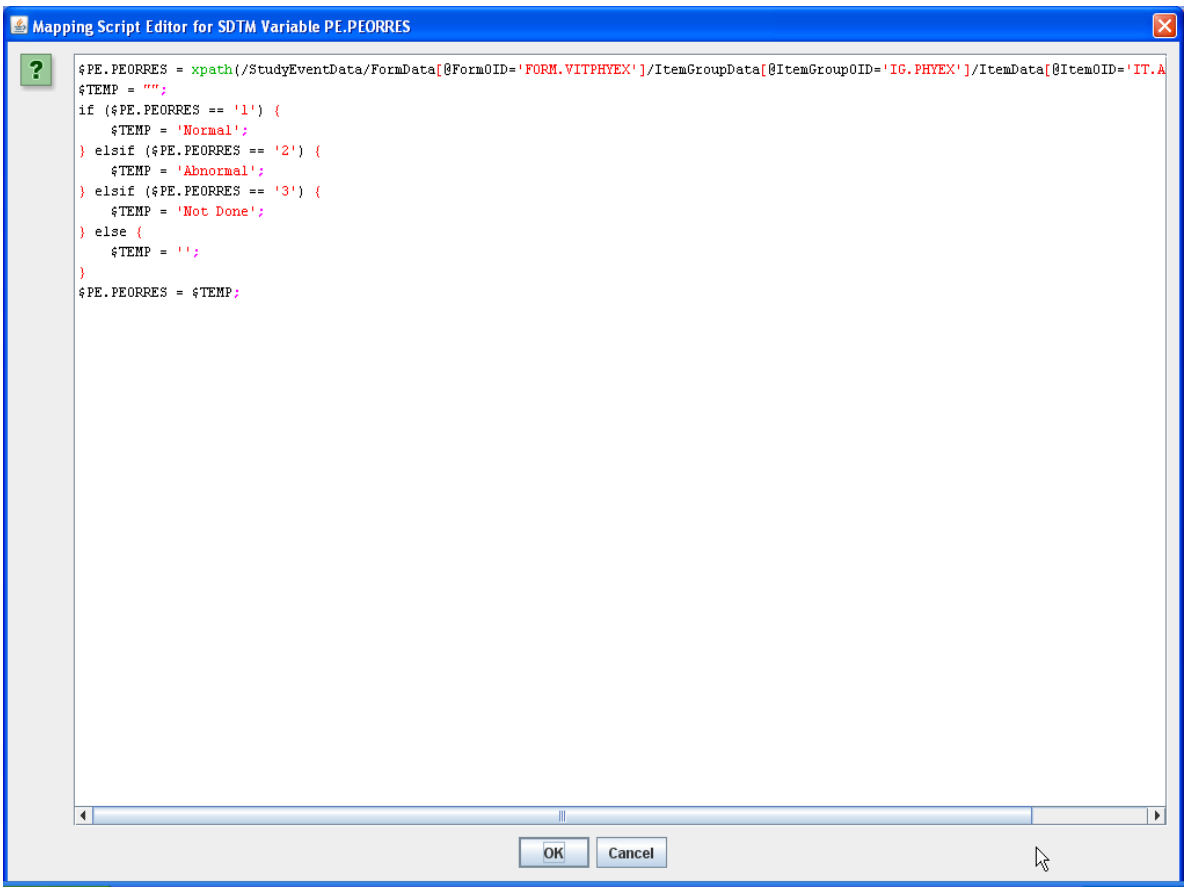
Larger text area for editing the mapping script

One of the users requested to have a larger text area for editing the mapping scripts.
This has now been implemented in two ways.

First of all, one can blend out the upper panel (with the selection comboboxes for StudyEvents, Forms, ItemGroups ...) by using the button “Hide upper panel”. This results in a larger area for editing the mapping script. At the same time, the button that was clicked changes into “Show upper panel”, allowing to restore the original view.



The second way of obtaining a larger editing window is by clicking the “Full.screen Transformation Script Panel” button. When clicked, a new window is opened which almost covers the full screen. The existing mapping script is copied into this new window, which eases further editing of complex mapping scripts.



When the "OK" button is then clicked, the window closes and the mapping script is copied into the normal editing panel. When "Cancel" is clicked, the window closes but the mapping script in the large window is not transferred to the normal editing panel, and the original mapping script is maintained.

Automated creation and population of the SDTM Comments (CO) domain

In the new version 1.4 it is now possible to automatically generate a CO domain which is similar to the automated creation of SUPPQUAL domains.

Essentially, the SDTM Comments domain is a result of the limitations of SAS Transport 5. Whereas one would expect that comments are just another field in each of the domains, the SDS development choose to make this a separate domain, as in SAS Transport 5 it is not possible to "flag" a field, except by giving a specific variable name[8].

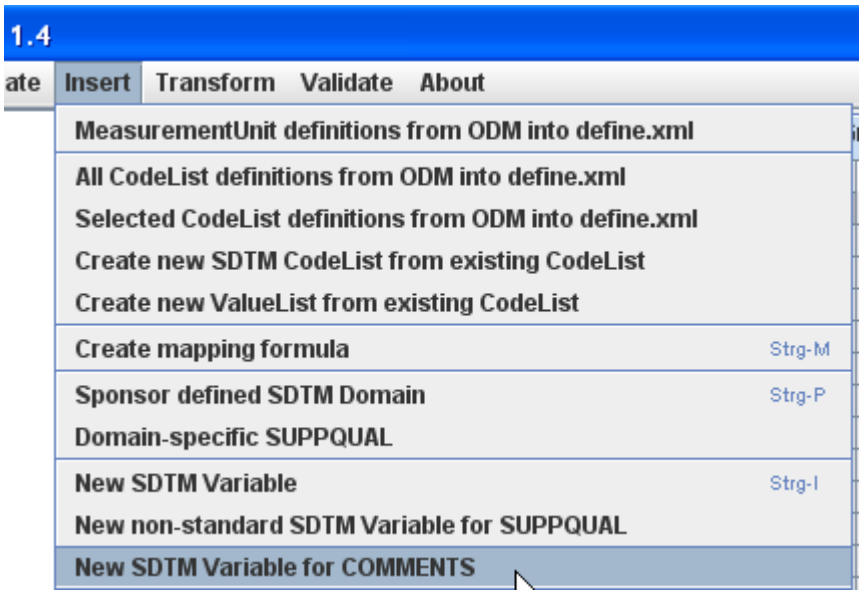
In ODM however, comments are either captured using a specific data item, or by using the Annotation/Comment child elements of ItemData. In the latter case, the comment belongs to the data point itself, and not to the CRF as a whole.

As of SDTM-ETL v.1.3, non-standard variables can be created. Usually these then go into a SUPPQUAL domain, such as SUPPDM, SUPPAE etc.
New in SDTM-ETL v.1.4 is that such a variable can be given the role of "COMMENT". If done so, the data will not flow into a SUPPQUAL domain, but a CO domain will be created and populated

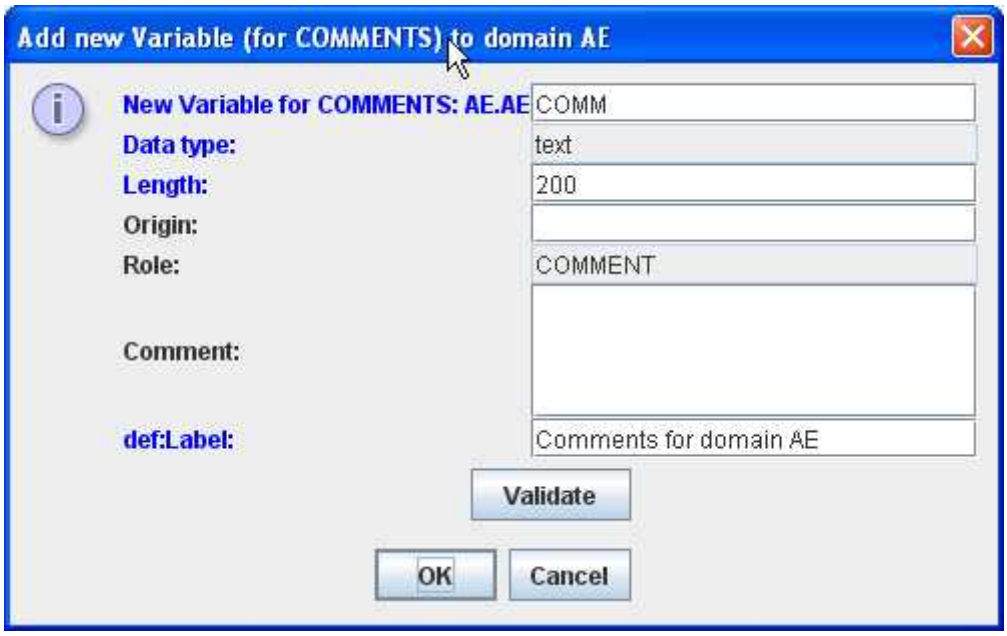
8 This could have been a good solution, e.g. to allow each domain to have a variable like AECOMM, PECOMM etc.

automatically.

In order to create a non-standard variable for which the data need to flow into the CO domain, use the menu “Insert -> New SDTM Variable for COMMENTS”:



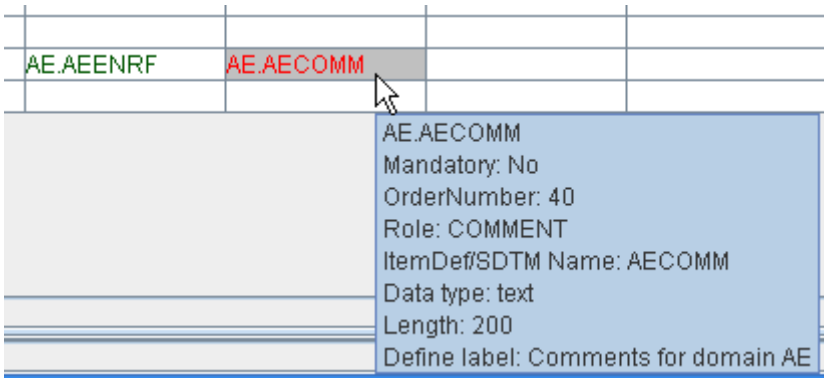
A dialog then is displayed allowing to add the necessary information:



The variable name is prepopulated with XX.XXCOMM, where “XX” is the domain name, in this case (for the AE) domain it is thus prepopulated with “AE.AECOMM”. The user is however free to choose another name.
The datatype is fixed to “text”, and a value of “200” is prepopulated.
One should provide a good estimate of the maximal length expected for comments. If the comments are expected to have a length of more than 200, the software will take care than sufficient fields (COVAL, COVAL1, COVAL2, etc.) will be created in the CO domain. However, for the creation of the “cleaned” define.xml, the software will need to know how many “COVALn” (“COVAL1”,

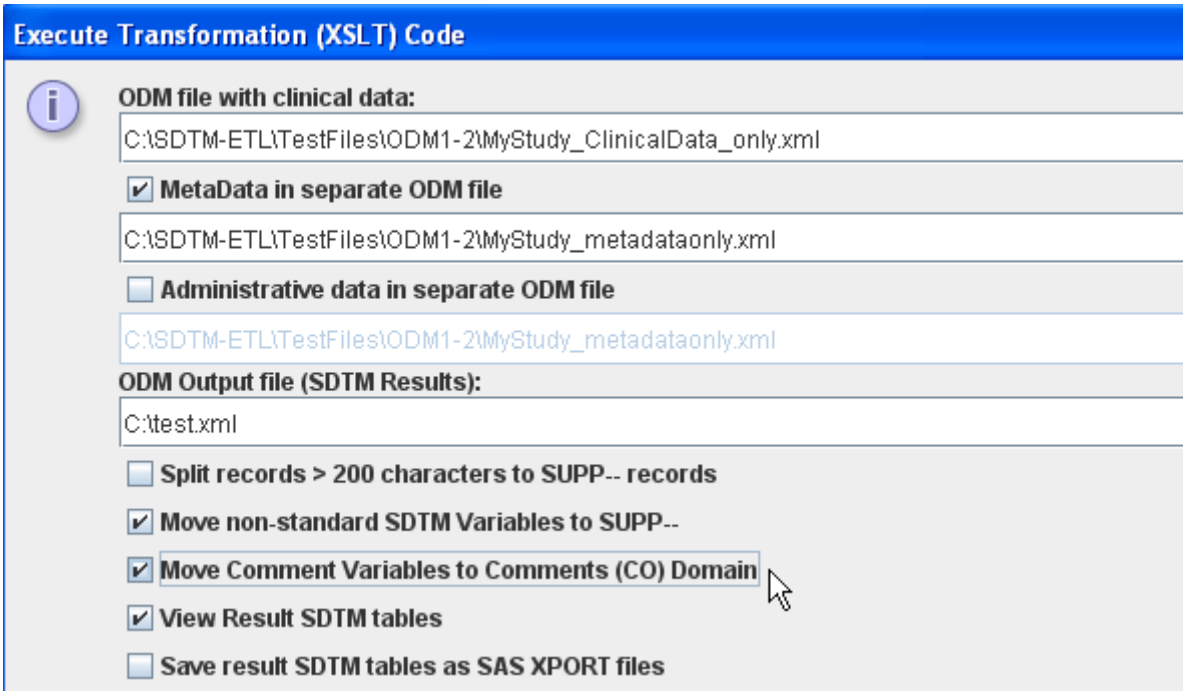
"COVAL2", etc.) variables will need to be declared in the define.xml export.
Setting a higher value for "Length" can also be of importance if one later wants to generate a relational SDTM database (see further).
The "Role" is fixed to "COMMENT". This is the flag for the software that the results of the transformation need to be moved to an instance of the CO domain.
Optional fields are the "Origin" and "Comment" fields. Mandatory is however to populate the "def:Label" field.

One can then use the "Validate" button to check whether all necessary fields are populated. When clicking the "OK" button, a short validation is performed, and when everything ok, a new non-standard SDTM variable is created as last in the row of the selected domain. It is colored red, designating it is a non-standard variable.



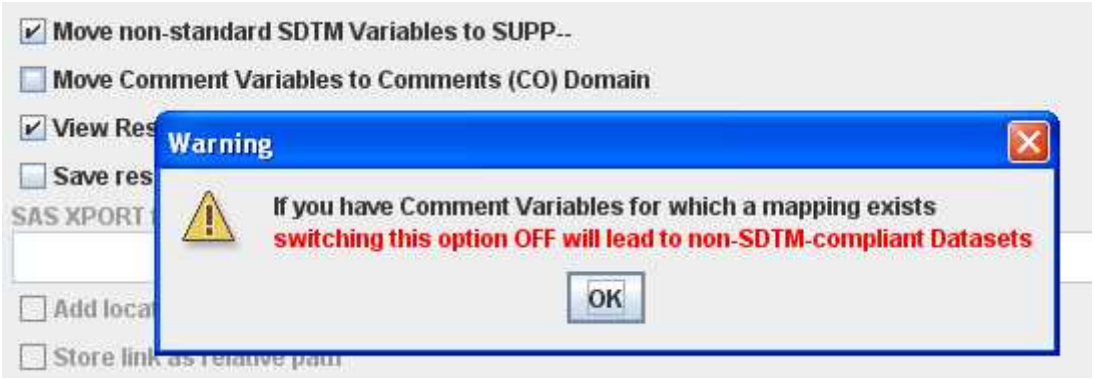
The mapping can then be defined as for any other variable.

When the mapping is executed, and there is a variable with role "COMMENT", then the following dialog will be displayed:



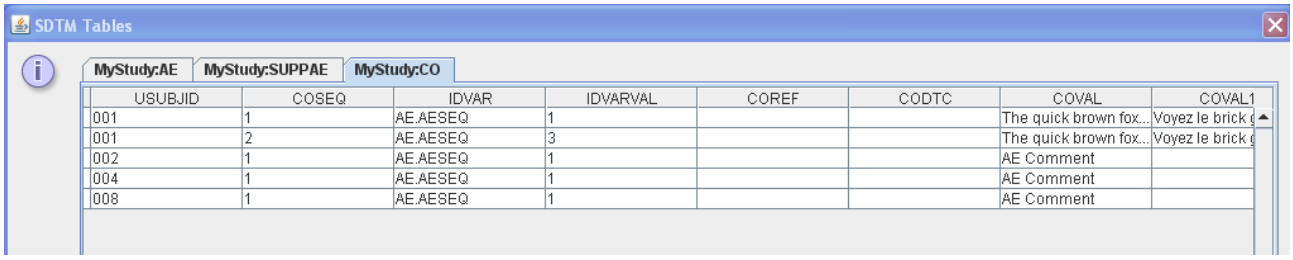
Remark the extra checkbox "*Move Comment Variables to Comments (CO) Domain*".

By default, it is checked. If one unchecks it, the following warning message is displayed:



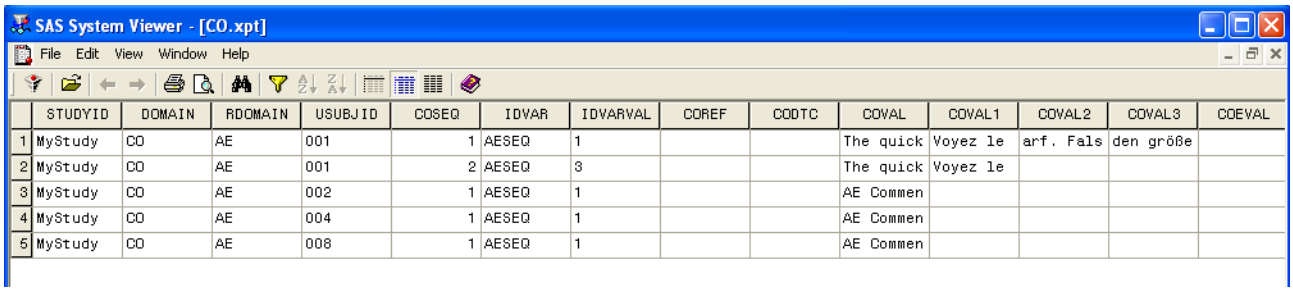
and a CO domain instance will not be generated automatically.

The result of such a transformation with automated generation of a CO domain instance will then e.g. be:



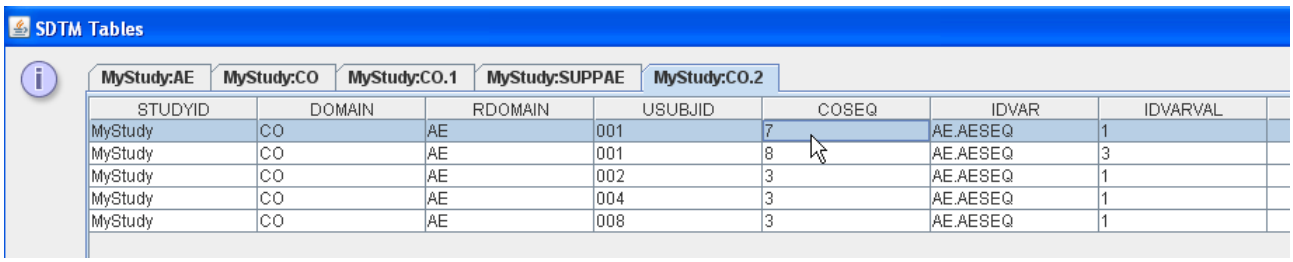
USUBJID	COSEQ	IDVAR	IDVARVAL	COREF	CODTC	COVAL	COVAL1
001	1	AE.AESEQ	1			The quick brown fox...	Voyez le brick
001	2	AE.AESEQ	3			The quick brown fox...	Voyez le brick
002	1	AE.AESEQ	1			AE Comment	
004	1	AE.AESEQ	1			AE Comment	
008	1	AE.AESEQ	1			AE Comment	

and if SAS XPT records were created:



	STUDYID	DOMAIN	RDOMAIN	USUBJID	COSEQ	IDVAR	IDVARVAL	COREF	CODTC	COVAL	COVAL1	COVAL2	COVAL3	COEVAL
1	MyStudy	CO	AE	001	1	AESEQ	1			The quick	Voyez le	arf. Fals	den größe	
2	MyStudy	CO	AE	001	2	AESEQ	3			The quick	Voyez le			
3	MyStudy	CO	AE	002	1	AESEQ	1			AE Commen				
4	MyStudy	CO	AE	004	1	AESEQ	1			AE Commen				
5	MyStudy	CO	AE	008	1	AESEQ	1			AE Commen				

In case the user has already created one or more instances of a CO domain, the software will automatically create an extra instance of the CO domain, taking into account the already existing instances. For example, if the user already created instances “CO” and “CO.1”, then the automatically created new instance will be “CO.2”.
Also, the sequence numbers will be synchronized among the different instances of the CO domain, at least when the user has checked the box “Sequence numbers unique across splitted domains” when asked for.
For example, the results for the automatically created domain instance “CO.2” when already a domain “CO” and “CO.1” have been created, will look like:



Remarks:

- When the comments are longer than 200 characters, then automatically, variables “COVAL1”, “COVAL2” etc. will be generated, and the content of the comments will be splitted into snippets of 200 characters long. Unlike for “SUPPQUAL” records, the splitting is not done between words, as the SDTM-IG does not require this.

- The generation of “COVAL1”, “COVAL2” etc. fields is also done even when the user has unchecked the box “Split records > 200 characters to SUPP-- records”. The reason for this is that generating a separate CO domain automatically only makes sense in case SAS Transport 5 records are created.

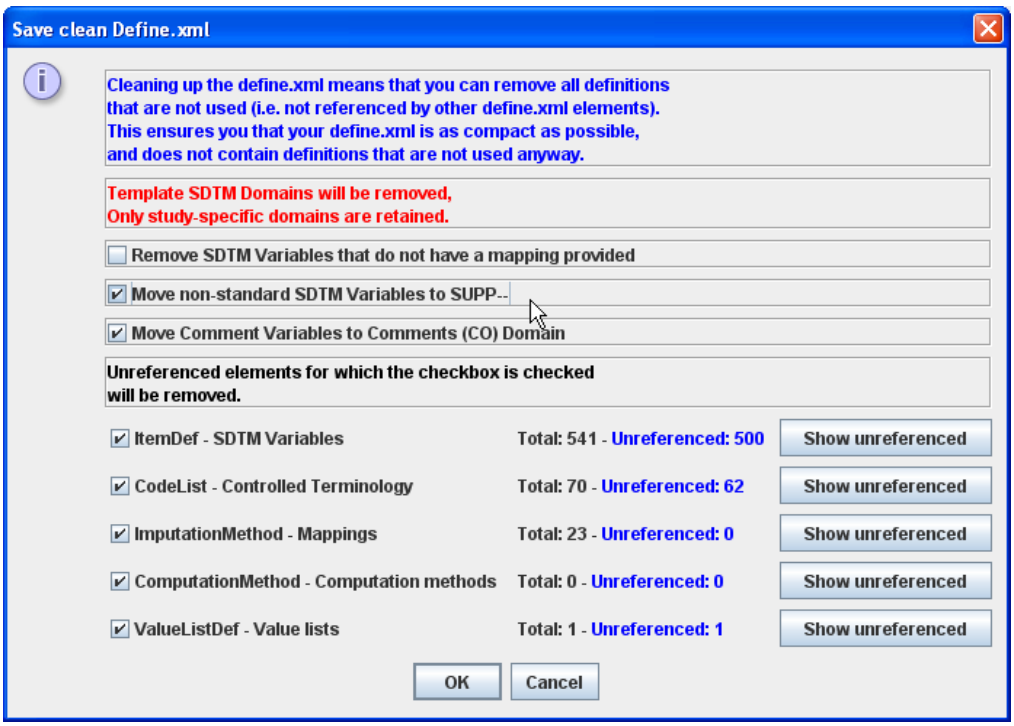
- When executing the functionality “Create SQL to generate Database tables”, no “CO” database table is currently created. The reason is (just as before) that we consider the automated generation of a CO domain as a feature to eliminate the limitations of SAS Transport 5.
So, when generating the SQL for generating an SDTM database, the “COMMENT” (e.g. AE.AECOMM) will be treated as just another SDTM variable.
In that case, it will be of importance to set a sufficient “Length” for the variable (when creating it or editing the variable properties) in order that a sufficiently long SQL field is created in the database to capture the complete record.

- When automatically generating a CO domain from “COMMENT” records, the CO-fields “COREF”, “CODTC” and “COEVAL” are not populated.

- Remark that the CO domain has a slightly different structure in SDTM 1.2 (SDTM-IG 3.1.2) than it has in SDTM 1.1 (SDTM-IG 3.1.1). This has been taken into account in the software.

Additional checkbox in dialog for “Save cleaned define.xml”

In order to support the automated generation of a CO domain from “COMMENT” records (see above), an extra checkbox “Move Comment Variables to Comments (CO) Domain” has been added to the dialog that is displayed when the user uses the menu “File -> Save cleaned define.xml”:



By default, this checkbox is checked, as the goal of this dialog is to generate a "near-submission-ready" define.xml for submission to the FDA, who still requires the data to come as SAS Transport 5 (XPT) files, meaning that comments must go into a separate CO.xpt file.

In case the checkbox is checked, and "COMMENT" variables are present in the mappings, the metadata for a "CO" domain are created, taking it into account when already other instances of the CO domain have been created, and also taking into account that when the maximal length of the comment exceeds 200 characters, additional variables COVAL1, COVAL2, etc. must be generated.

New feature: the "comments()" function

A new function has been added to the build-in library of SDTM-ETL functions.
The "comments()" function takes an ItemData element as argument, and then looks for the (text) value of the underlying Annotation/Comment element.

So if in the ODM ClinicalData one has:

```
- <ItemData ItemOID="IT.TAREA" Value="ONC">
  - <Annotation SeqNum="1" TransactionType="Insert">
      <Comment SponsorOrSite="Site">Site Comment</Comment>
    - <Flag>
        <FlagValue CodeListOID="CL.TAREAF">FV1</FlagValue>
        <FlagType CodeListOID="CL.TAREAF">FT1</FlagType>
      </Flag>
    - <Flag>
        <FlagValue CodeListOID="CL.TAREAF">FV2</FlagValue>
        <FlagType CodeListOID="CL.TAREAF">FT2</FlagType>
      </Flag>
    </Annotation>
  </ItemData>
```

The function “comment()” will deliver the text “Site Comment” when applied on the Item with OID “IT.TAREA”.

This new function is of course especially interesting in combination with the functionality to automatically create a Comments (CO) domain for non-standard SDTM variables that have the role “COMMENT” (see before).

Remark that the function will only work on an ItemData element, not on an ItemData Value.
So if your XPath expression (e.g. by drag-and-drop) is:

$TEMP =
xpath(/StudyEventData/FormData[@FormOID='FORM.AE']/ItemGroupData[@ItemGroupOID='IG.AE']/ItemData[@ItemOID='IT.AETERM']/@Value);

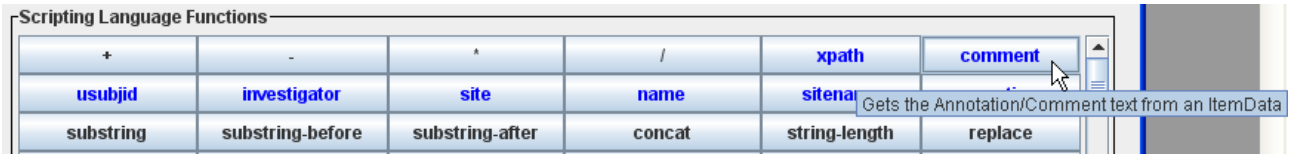
you will need to (manually) remove the “/@Value” part, in order to have:

$TEMP =
xpath(/StudyEventData/FormData[@FormOID='FORM.AE']/ItemGroupData[@ItemGroupOID='IG.AE']/ItemData[@ItemOID='IT.AETERM']);

One can then apply the function, e.g.:

$AE.AECOMM = comment($TEMP);

The new function is also listed in the dialog for defining the mapping as an extra button:



The current limitation of this new function is that it only takes the first “Annotation/Comment” that has been added to an ItemData element, i.e. the treatment of multiple annotations to a single ODM data point is not supported yet.

Support for the draft Pharmacogenomics (PG) and Pharmacogenomics Findings (PF) domains

The distribution contains a template define.xml file (in the directory “define”) “define_template_SDTM_1.2_PGx.xml” which also contains the variables for the new Pharmacogenomics (PG) and Pharmacogenomics Findings (PF) domains.
These two domains have recently been published “for public comment”, so they are “draft” do not belong to the SDTM standard (1.2) yet.
If you would like to use these two domains anyway, please rename the file “define_template_SDTM_1.2.xml” to something else (e.g. “define_template_SDTM_1.2_orig.xml) and rename the file “ define_template_SDTM_1.2_PGx.xml” to “ define_template_SDTM_1.2.xml”.
If done so, these two draft domains will be loaded together with the original ones when the user selects to work with “SDTM 1.2” when starting on a set of new mappings.

Other changes to the template define.xml files

When the define.xml standard (or better "CRT-DDS") standard was published, it was not 100% clear what the value for the "Name" attribute on "ItemGroupDef" should be. The specification stated "*The filename of the dataset or data domain name*", but did not state that this needs to be the (usually two-character) domain code.
The ODM specification (on which define.xml is based) states "*any sequence of characters*" and "*A name is intended to be a human readable name for some entity*".
Later, when the (draft) "MetaData Implementation Guides" were published, it became clear that it is expected that the domain code is used for the "Name" attribute in "ItemGroupDef".

In our previous version, for the SDTM-1.1 templates we e.g. used "Demographics" for the value for the "Name" attribute. For the SDTM-1.2 templates, we did already use the domain code (e.g. "DM").
In version 1.4, the template define.xml file for SDTM-1.1 has been corrected, and the domain code is now also used for the "Name" attribute of the "ItemGroupDef" elements.