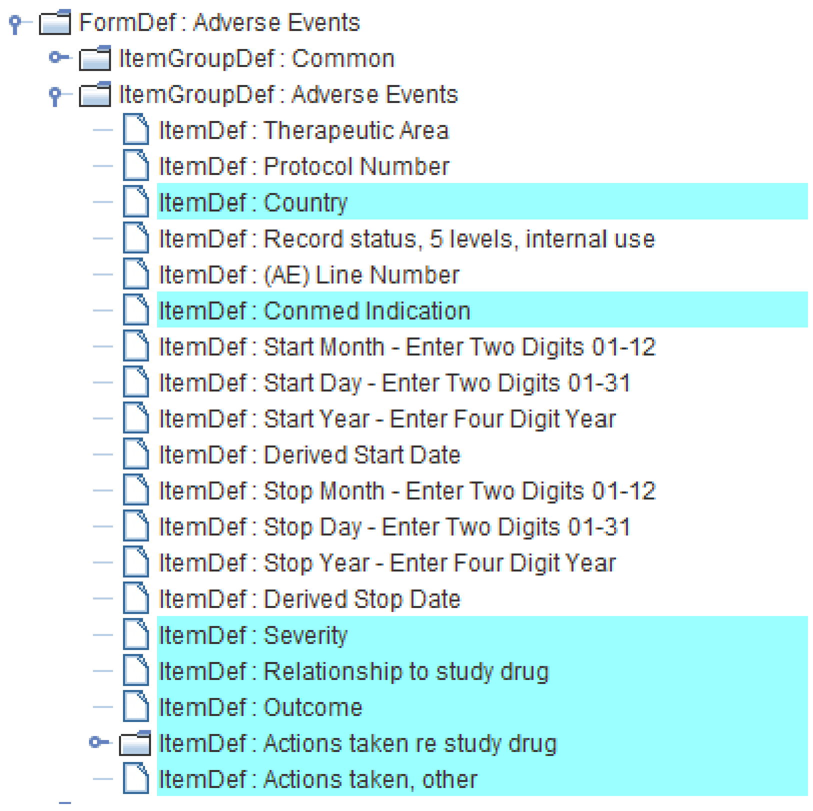
# SDTM-ETL 4.x User Manual and Tutorial

Author: Jozef Aerts, XML4Pharma

Last update: 2022-06-22

## Creating mappings for the AE domain

Now that we have created (and executed) mappings for several domains, let us create a mapping for the Adverse Events (AE) domain. If we take a look at our ODM study setup, we notice that there is an "Adverse Events" form, which is used (optionally) in the second visit.
Usually, Adverse Event forms can be used in any visit, so we will do as if there were more visits defined in this study, and allow Adverse Event forms to have been used in any of the visits (StudyEvents).

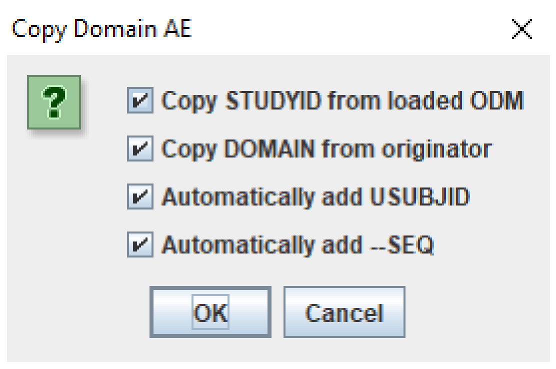


FormDef : Adverse Events
ItemGroupDef : Common
ItemGroupDef : Adverse Events
ItemDef : Therapeutic Area
ItemDef : Protocol Number
ItemDef : Country
ItemDef : Record status, 5 levels, internal use
ItemDef : (AE) Line Number
ItemDef : Conmed Indication
ItemDef : Start Month - Enter Two Digits 01-12
ItemDef : Start Day - Enter Two Digits 01-31
ItemDef : Start Year - Enter Four Digit Year
ItemDef : Derived Start Date
ItemDef : Stop Month - Enter Two Digits 01-12
ItemDef : Stop Day - Enter Two Digits 01-31
ItemDef : Stop Year - Enter Four Digit Year
ItemDef : Derived Stop Date
ItemDef : Severity
ItemDef : Relationship to study drug
ItemDef : Outcome
ItemDef : Actions taken re study drug
ItemDef : Actions taken, other

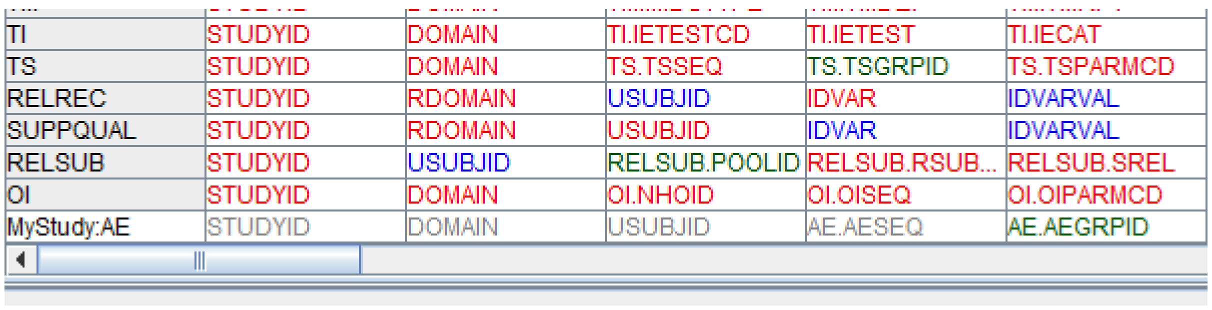
A further look at the Adverse Event form shows that it is only partially suited for mapping with the SDTM. Ideally, the CDASH[1] AE form should have been used, as it has an (almost) 1:1 mapping with the SDTM AE domain, but at the time of the current study design, this CDASH form was not available yet. This will happen often in the case of legacy studies.

To start, drag-and-drop the AE row in the SDTM table to the bottom of the table, this creating a study-specific instance of the AE domain. Accept the defaults for automatic generation of mappings for STUDYID, DOMAIN, USUBJID and AE.AESEQ:

[1]See the CDISC website at http://www.cdisc.org/standards/cdash/index.html

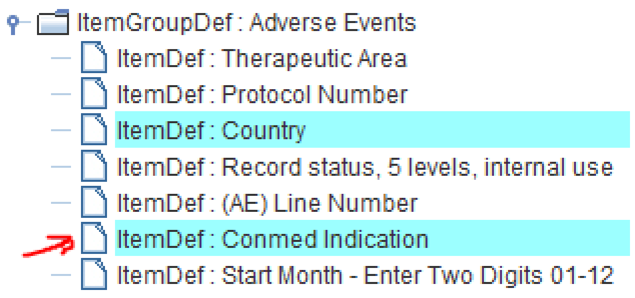

Copy Domain AE

- [✓] Copy STUDYID from loaded ODM
- [✓] Copy DOMAIN from originator
- [✓] Automatically add USUBJID
- [✓] Automatically add --SEQ

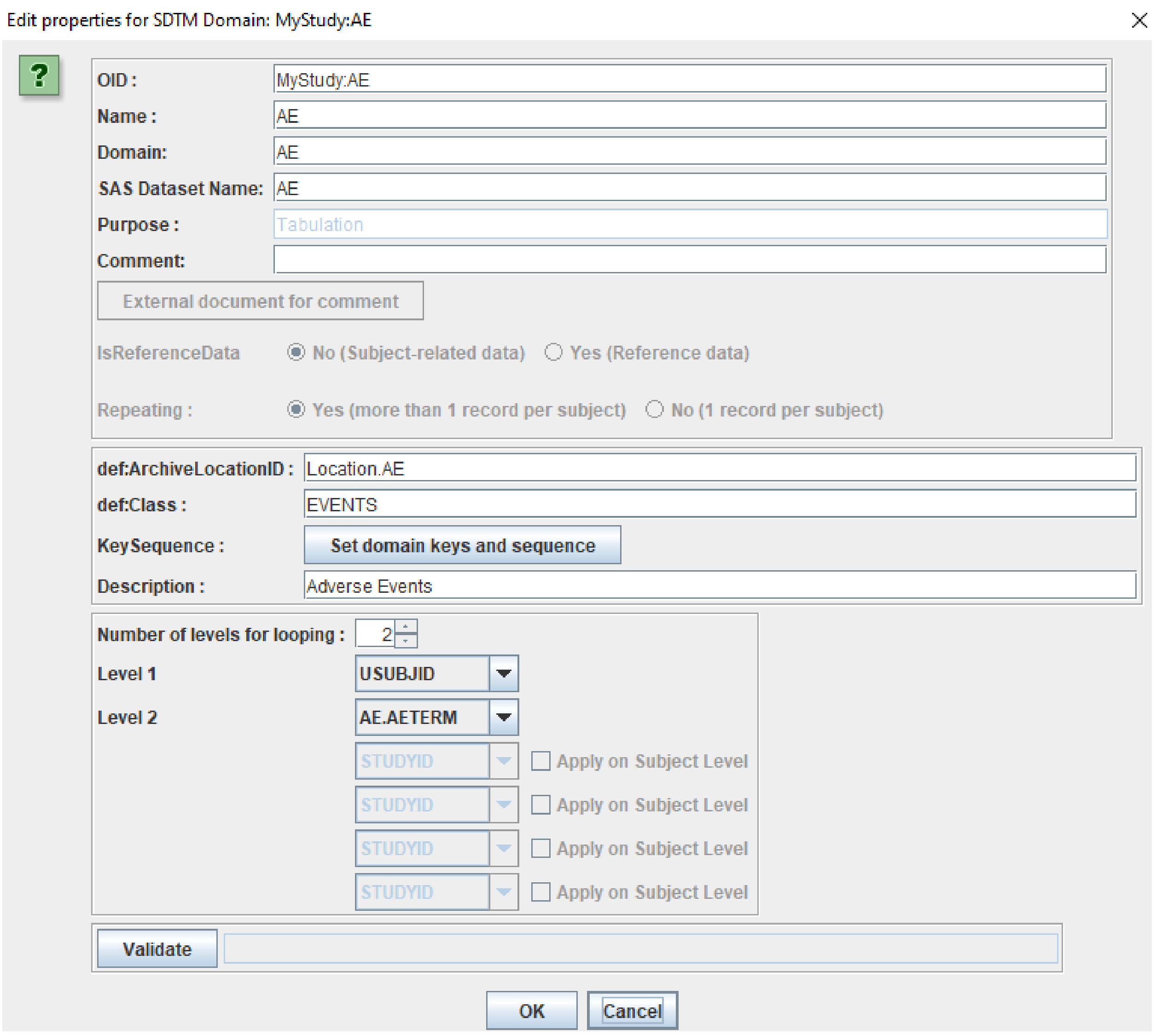


| | | | | | |
|---|---|---|---|---|---|
| TI | STUDYID | DOMAIN | TI.IETESTCD | TI.IETEST | TI.IECAT |
| TS | STUDYID | DOMAIN | TS.TSSEQ | TS.TSGRPID | TS.TSPARMCD |
| RELREC | STUDYID | RDOMAIN | USUBJID | IDVAR | IDVARVAL |
| SUPPQUAL | STUDYID | RDOMAIN | USUBJID | IDVAR | IDVARVAL |
| RELSUB | STUDYID | USUBJID | RELSUB.POOLID | RELSUB.RSUB... | RELSUB.SREL |
| OI | STUDYID | DOMAIN | OI.NHOID | OI.OISEQ | OI.OIPARMCD |
| MyStudy:AE | STUDYID | DOMAIN | USUBJID | AE.AESEQ | AE.AEGRPID |

The SDTM Implementation Guide tells us that we should have one record per adverse event per subject. We can translate this into "One record per AE.AETERM per USUBJID", this as each ODM record has an "IT.AETERM" Item with the name "Conmed Indication" in the form „Adverse Events". Please note that also other choices are possible, as long as the choice ensures that there is one record per AE form/subform used. We do however choose for AETERM as it is the SDTM topic variable.


ItemGroupDef : Adverse Events
- ItemDef : Therapeutic Area
- ItemDef : Protocol Number
- ItemDef : Country
- ItemDef : Record status, 5 levels, internal use
- ItemDef : (AE) Line Number
- ItemDef : Conmed Indication
- ItemDef : Start Month - Enter Two Digits 01-12

Double click the first cell in our newly created row ("MyStudy:AE"). The Domain Editor panel shows up:


Edit properties for SDTM Domain: MyStudy:AE

| | |
|---|---|
| OID : | MyStudy:AE |
| Name : | AE |
| Domain: | AE |
| SAS Dataset Name: | AE |
| Purpose : | Tabulation |
| Comment: | |

External document for comment

IsReferenceData [✓ No (Subject-related data)] Yes (Reference data)

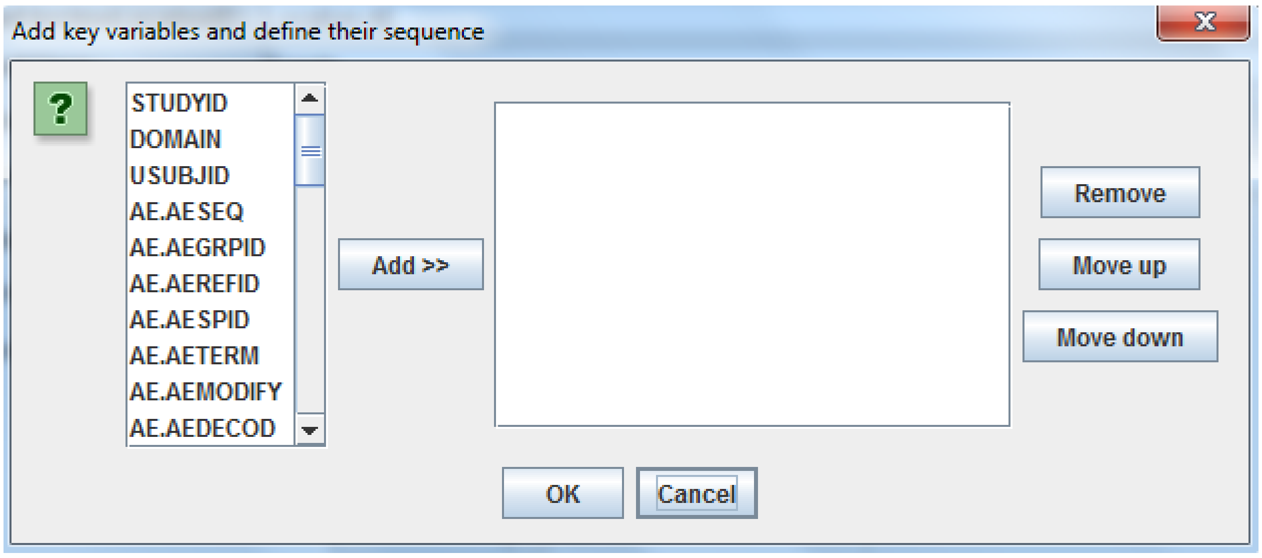
Repeating : [✓ Yes (more than 1 record per subject)] No (1 record per subject)

| | |
|---|---|
| def:ArchiveLocationID : | Location.AE |
| def:Class : | EVENTS |
| KeySequence : | Set domain keys and sequence |
| Description : | Adverse Events |

Number of levels for looping : 2

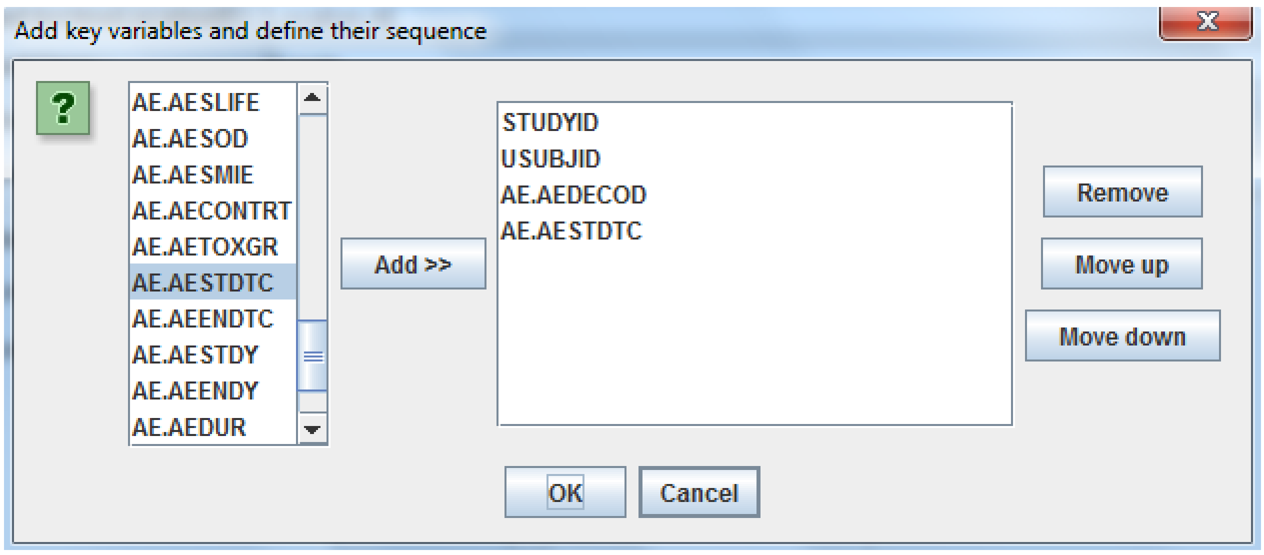
| | | |
|---|---|---|
| Level 1 | USUBJID | |
| Level 2 | AE.AETERM | |
| | STUDYID | Apply on Subject Level |
| | STUDYID | Apply on Subject Level |
| | STUDYID | Apply on Subject Level |
| | STUDYID | Apply on Subject Level |

We select two levels of looping, the second being AE.AETERM.
If we use define.xml 2.0 or 2.1, there is also a button „Set domain keys and sequence". Clicking it pops up the dialog:


Add key variables and define their sequence

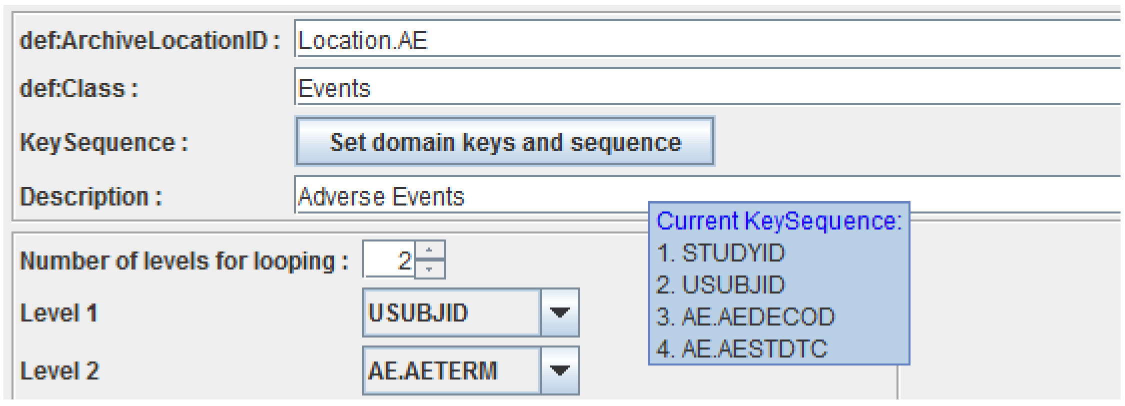
STUDYID
DOMAIN
USUBJID
AE.AESEQ
AE.AEGRPID
AE.AEREFID
AE.AESPID
AE.AETERM
AE.AEMODIFY
AE.AEDECOD

You can now add some variables using the „Add>>" button and change the order with the „Move up" and „Move down" buttons. In earlier days, the SDTM-IG gave some suggestions for the keys, but most sponsors just copied these without further thinking[2]. You might e.g. choose for:



STUDYID
USUBJID
AE.AEDECOD
AE.AESTDTC

The next time you will edit the domain properties and hold the mouse over the button, you will find:



def:ArchiveLocationID : Location.AE
def:Class : Events
KeySequence :
Description : Adverse Events
Number of levels for looping : 2
Level 1 USUBJID
Level 2 AE.AETERM

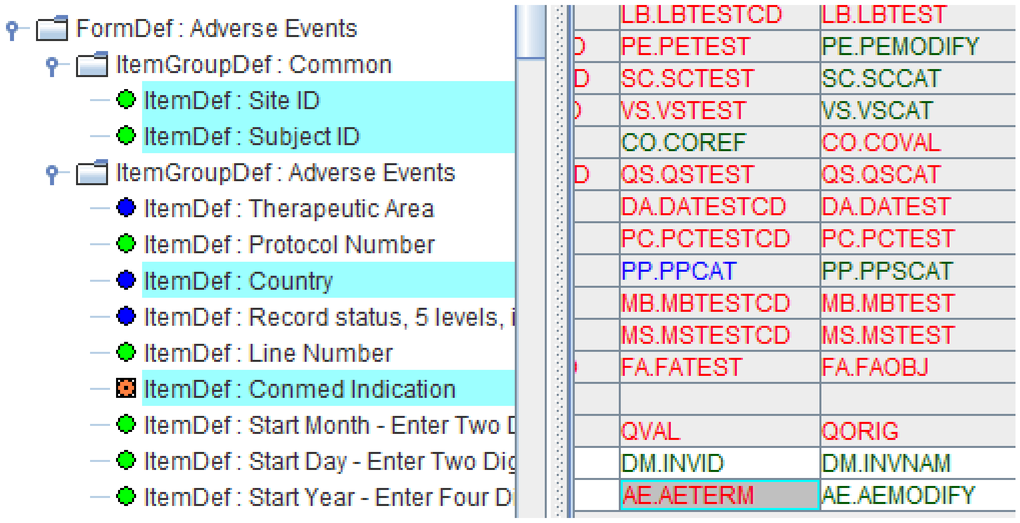
Current KeySequence:
1. STUDYID
2. USUBJID
3. AE.AEDECOD
4. AE.AESTDTC

After having edited the domain properties, we can start with the mappings.

We do already have a mapping for USUBJID (automatically generated), so we must still generate a mapping for AE.AETERM. It is always good practice to first develop mappings for those SDTM variables that are looping variables (usually that is the topic variable), so let us do this one first. As already stated the ODM has an Item "Conmed Indication" with OID "IT.AETERM". When we click in the SDTM cell for AE.AETERM, we even see that this Item is a "hot candidate" for the mapping (meaning that it has an annotation that its SDTM target[3] is "AETERM"):

[2]It is however also not clear whether the FDA is doing anything with this information! The define.xml 2.0 specification states that adding domain keys in the context of a regulatory submission. To our knowledge, the FDA is not loading submission datasets in a database, so we wonder why these keys are necessary...

[3]In the ODM, this is either accomplished by the „SDSVarName" attribute on ItemDef, or by an „Alias" element having „SDTM" for the „Context" attribute.

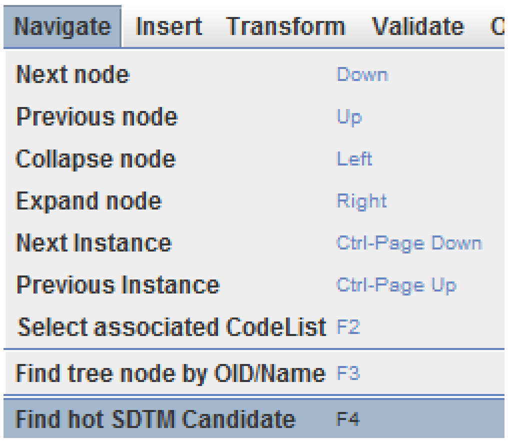


- FormDef : Adverse Events
  - ItemGroupDef : Common
    - ● ItemDef : Site ID
    - ● ItemDef : Subject ID
  - ItemGroupDef : Adverse Events
    - ● ItemDef : Therapeutic Area
    - ● ItemDef : Protocol Number
    - ● ItemDef : Country
    - ● ItemDef : Record status, 5 levels, i
    - ● ItemDef : Line Number
    - ● ItemDef : Conmed Indication
    - ● ItemDef : Start Month - Enter Two D
    - ● ItemDef : Start Day - Enter Two Dig
    - ● ItemDef : Start Year - Enter Four D

| | |
|---|---|
| LB.LBTESTCD | LB.LBTEST |
| PE.PETEST | PE.PEMODIFY |
| SC.SCTEST | SC.SCCAT |
| VS.VSTEST | VS.VSCAT |
| CO.COREF | CO.COVAL |
| QS.QSTEST | QS.QSCAT |
| DA.DATESTCD | DA.DATEST |
| PC.PCTESTCD | PC.PCTEST |
| PP.PPCAT | PP.PPSCAT |
| MB.MBTESTCD | MB.MBTEST |
| MS.MSTESTCD | MS.MSTEST |
| FA.FATEST | FA.FAOBJ |
| | |
| QVAL | QORIG |
| DM.INVID | DM.INVNAM |
| AE.AETERM | AE.AEMODIFY |

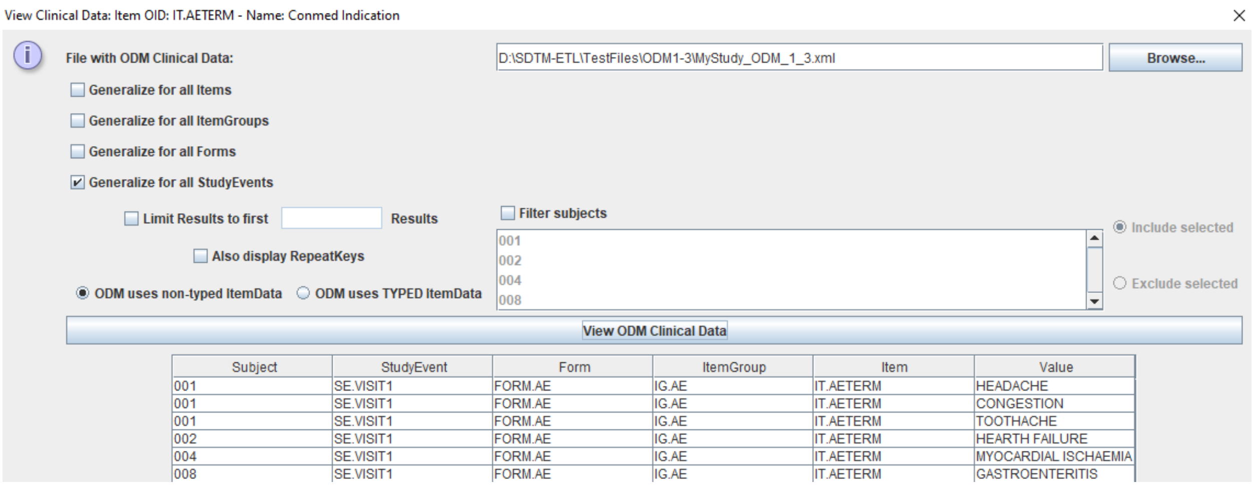
The square around the „traffic light" indicating that the item is a „hot candidate" for the SDTM variable „AETERM".

If the item in the tree is not visible yet, we also can always find the „hot candidate" using the menu „Navigate – Find hot SDTM Candidate„:



Navigate
Insert
Transform
Validate
Next node Down
Previous node Up
Collapse node Left
Expand node Right
Next Instance Ctrl-Page Down
Previous Instance Ctrl-Page Up
Select associated CodeList F2
Find tree node by OID/Name F3
Find hot SDTM Candidate F4

To verify whether this really is the information that needs to go into AETERM, after selecting "Conmed Indication", we can also use the menu "View - ODM Clinical Data", e.g. leading to:



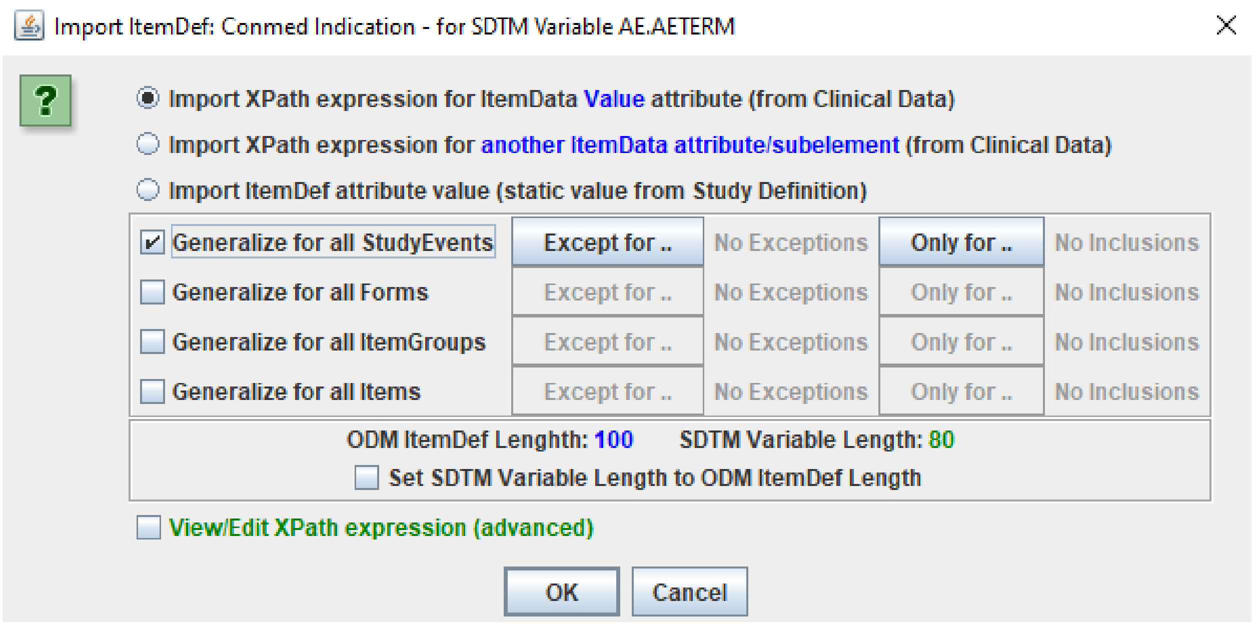
View Clinical Data: Item OID: IT.AETERM - Name: Conmed Indication

File with ODM Clinical Data: D:\SDTM-ETL\TestFiles\ODM1-3\MyStudy_ODM_1_3.xml

- [ ] Generalize for all Items
- [ ] Generalize for all ItemGroups
- [ ] Generalize for all Forms
- [✓] Generalize for all StudyEvents
- [ ] Limit Results to first Results
- [ ] Also display RepeatKeys
- [✓ ODM uses non-typed ItemData] ODM uses TYPED ItemData
- [ ] Filter subjects: 001, 002, 004, 008
- [✓ Include selected] Exclude selected

| Subject | StudyEvent | Form | ItemGroup | Item | Value |
|---|---|---|---|---|---|
| 001 | SE.VISIT1 | FORM.AE | IG.AE | IT.AETERM | HEADACHE |
| 001 | SE.VISIT1 | FORM.AE | IG.AE | IT.AETERM | CONGESTION |
| 001 | SE.VISIT1 | FORM.AE | IG.AE | IT.AETERM | TOOTHACHE |
| 002 | SE.VISIT1 | FORM.AE | IG.AE | IT.AETERM | HEARTH FAILURE |
| 004 | SE.VISIT1 | FORM.AE | IG.AE | IT.AETERM | MYOCARDIAL ISCHAEMIA |
| 008 | SE.VISIT1 | FORM.AE | IG.AE | IT.AETERM | GASTROENTERITIS |

Just drag-and-drop the Item „Conmed Indication" from the left side to the cell "AE.AETERM". The following dialog shows up:


Import ItemDef: Conmed Indication - for SDTM Variable AE.AETERM

(●) Import XPath expression for ItemData Value attribute (from Clinical Data)
( ) Import XPath expression for another ItemData attribute/subelement (from Clinical Data)
( ) Import ItemDef attribute value (static value from Study Definition)

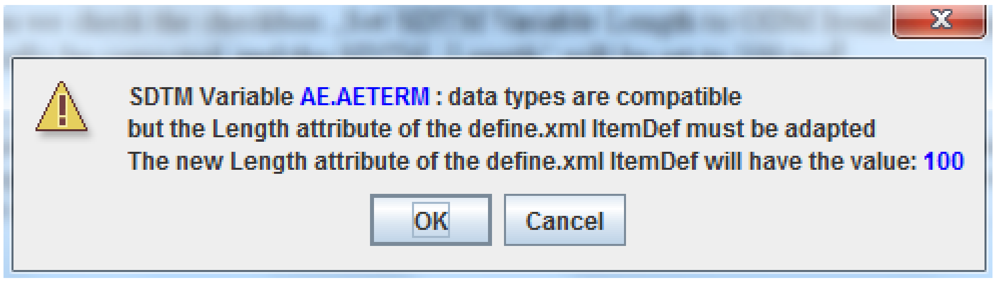
|  |  |  |  |  |
|---|---|---|---|---|
| [✓] Generalize for all StudyEvents | Except for .. | No Exceptions | Only for .. | No Inclusions |
| [ ] Generalize for all Forms | Except for .. | No Exceptions | Only for .. | No Inclusions |
| [ ] Generalize for all ItemGroups | Except for .. | No Exceptions | Only for .. | No Inclusions |
| [ ] Generalize for all Items | Except for .. | No Exceptions | Only for .. | No Inclusions |

ODM ItemDef Length: 100 SDTM Variable Length: 80
[ ] Set SDTM Variable Length to ODM ItemDef Length

[ ] View/Edit XPath expression (advanced)

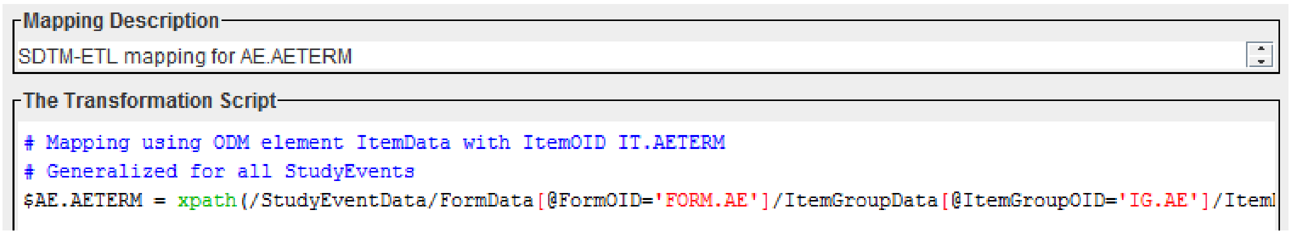
We accept the default that will retrieve the value from the clinical data („ItemData **Value** attribute") As we want to take care that AE forms from all visits are taken into account (although in this study it is only used in the second visit, as there are only two visits), we need to check the checkbox "Generalize for all StudyEvents". Later we will learn how to be more precise which visits exactly must be selected (unimportant in this case) using the „Except for ..." and „Only for ..." buttons. We are also warned that in the ODM, the value for „Conmed Indication" may be up to 100 characters long, whereas our define.xml template has a default of max. 80 characters for this variable. When we check the checkbox „Set SDTM Variable Length to ODM ItemDef Length", this will automatically be corrected, and the SDTM „Length" will be set to 100 too[4].

When clicking OK, this correction is again confirmed:


SDTM Variable AE.AETERM : data types are compatible
but the Length attribute of the define.xml ItemDef must be adapted
The new Length attribute of the define.xml ItemDef will have the value: 100

The transformation script is now automatically generated:

[4]It is NOT a good idea to set the length for all variables to the SAS-XPT maximum of 200, as this will generate extremely large SAS-XPT data sets that the FDA cannot handle. Also, setting a correct maximum length will help generating an optimized database (see "Generating an SDTM database")

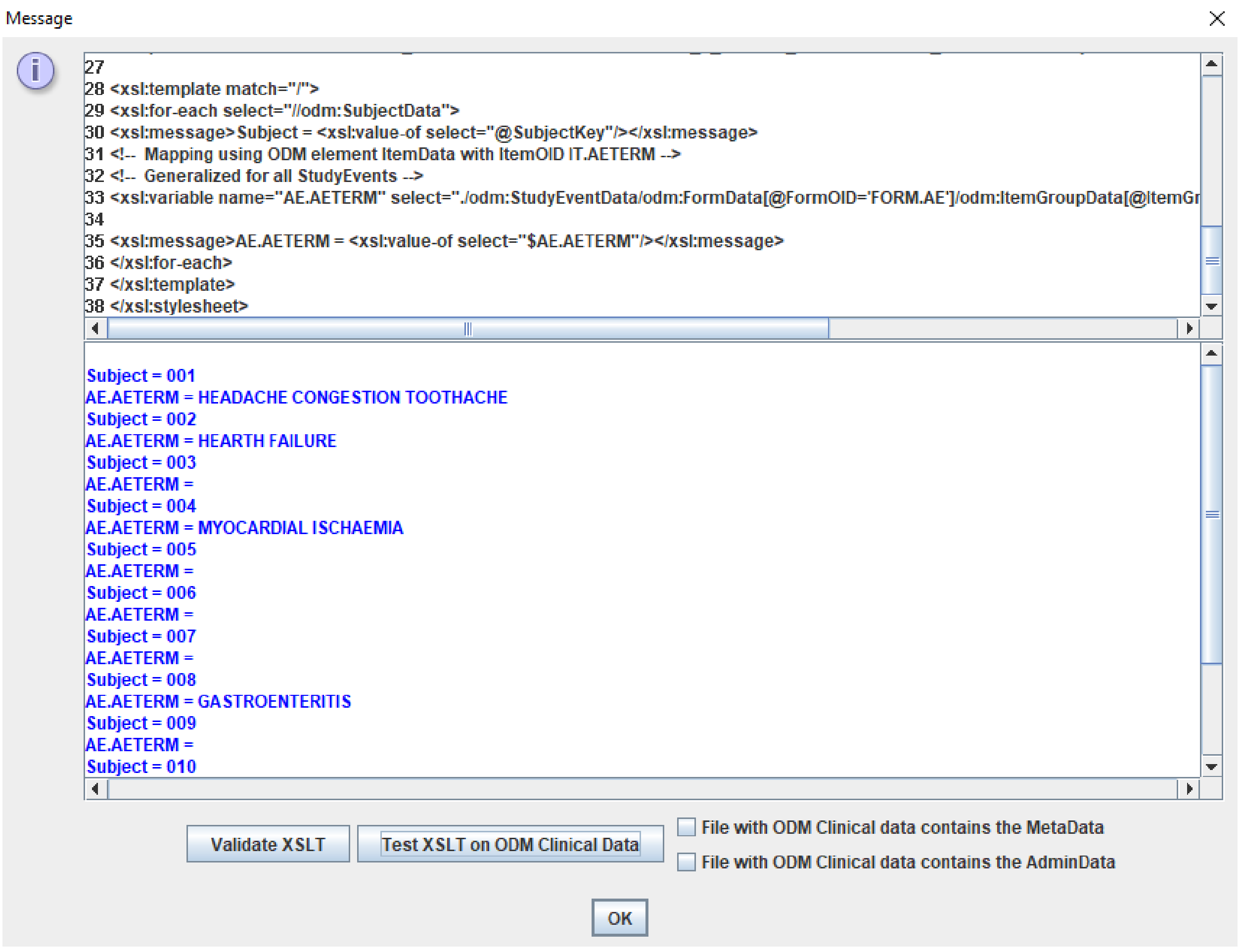


SDTM-ETL mapping for AE.AETERM

```
# Mapping using ODM element ItemData with ItemOID IT.AETERM
# Generalized for all StudyEvents
$AE.AETERM = xpath(/StudyEventData/FormData[@FormOID='FORM.AE']/ItemGroupData[@ItemGroupOID='IG.AE']/Iteml
```

A "mapping description" is automatically created, which you might want to adapt. The information in it will go into the „Description" element of the corresponding „MethodDef" element in the underlying define.xml (in case define.xml 2.0 is used).

We can test it using the button "Test – Transform to XSLT". We get something like:



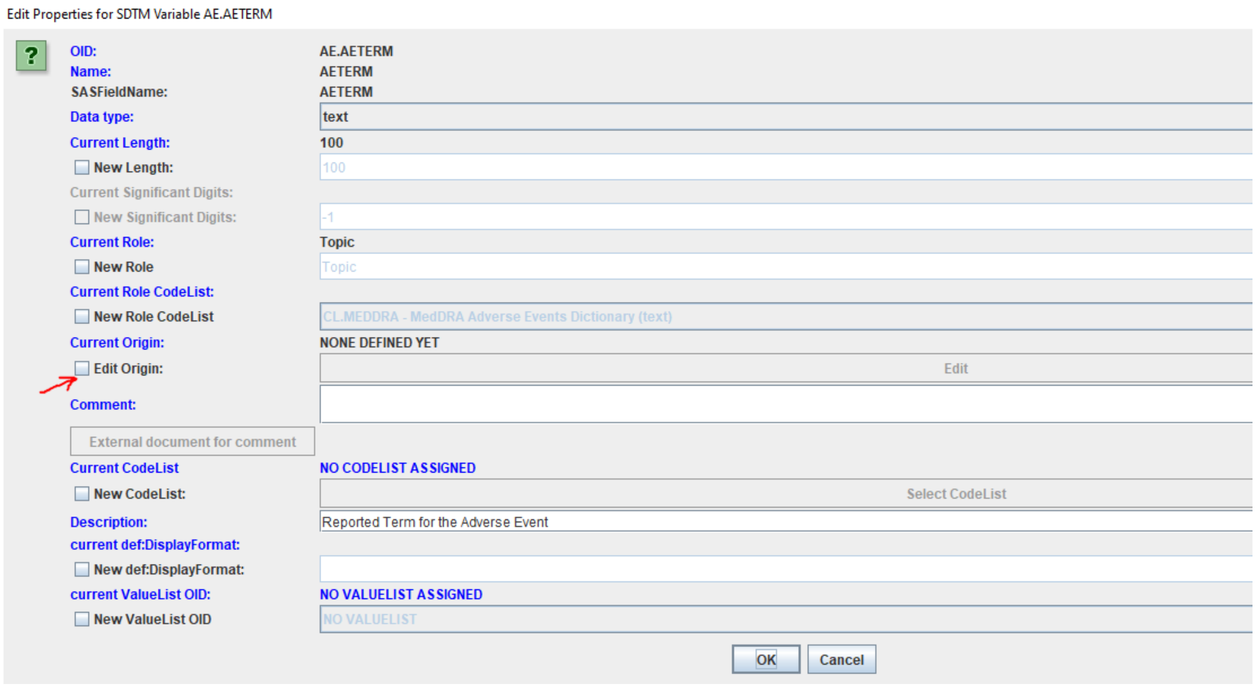
```
<xsl:template match="/">
<xsl:for-each select="//odm:SubjectData">
<xsl:message>Subject = <xsl:value-of select="@SubjectKey"/></xsl:message>
<!-- Mapping using ODM element ItemData with ItemOID IT.AETERM -->
<!-- Generalized for all StudyEvents -->
<xsl:variable name="AE.AETERM" select="./odm:StudyEventData/odm:FormData[@FormOID='FORM.AE']/odm:ItemGroupData[@ItemGr

<xsl:message>AE.AETERM = <xsl:value-of select="$AE.AETERM"/></xsl:message>
</xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

```
Subject = 001
AE.AETERM = HEADACHE CONGESTION TOOTHACHE
Subject = 002
AE.AETERM = HEARTH FAILURE
Subject = 003
AE.AETERM =
Subject = 004
AE.AETERM = MYOCARDIAL ISCHAEMIA
Subject = 005
AE.AETERM =
Subject = 006
AE.AETERM =
Subject = 007
AE.AETERM =
Subject = 008
AE.AETERM = GASTROENTERITIS
Subject = 009
AE.AETERM =
Subject = 010
```
Validate XSLT
Test XSLT on ODM Clinical Data
File with ODM Clinical data contains the MetaData
File with ODM Clinical data contains the AdminData
OK

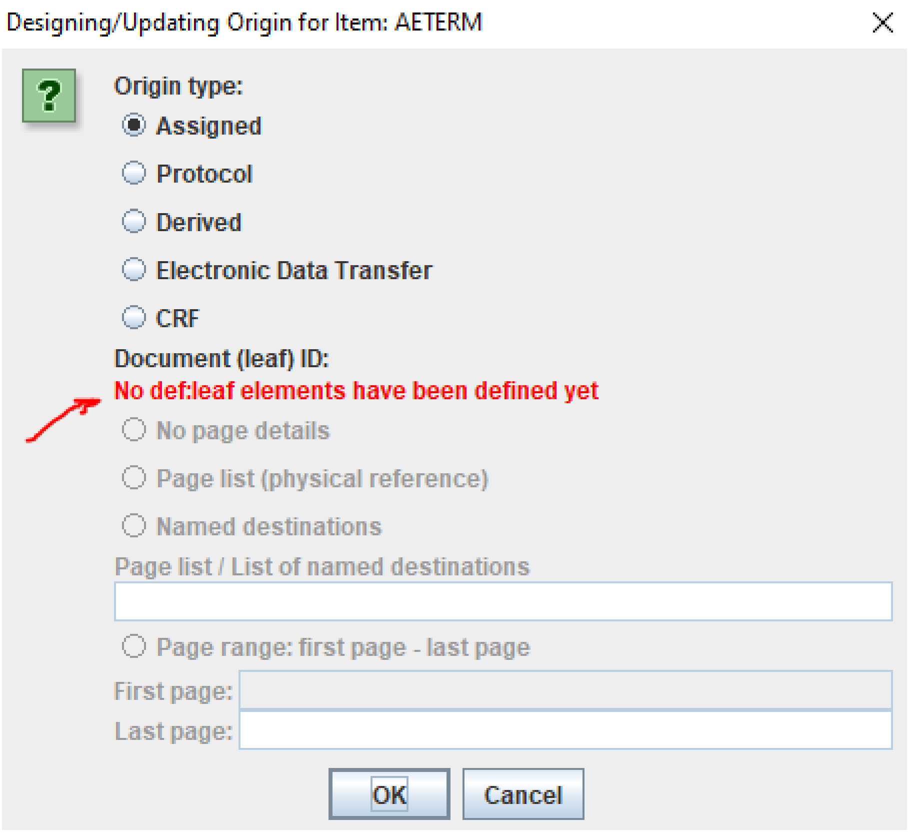
after having used the „Test XSLT on ODM Clinical Data" button and having provided a file with clinical data for this study. We also see that some subjects do not have any adverse events (e.g. subjects 005, 006, 007). When later executing the mappings, no AE records will be generated for these subjects.

Define.xml 2.0 also requires to provide the „Origin" of each SDTM variable when the context is a regulatory submission. To add it for AETERM, use the menu „Edit – SDTM Variable Properties" and then look for the checkbox "Edit Origin":


Edit Properties for SDTM Variable AE.AETERM

| | |
|---|---|
| OID: | AE.AETERM |
| Name: | AETERM |
| SASFieldName: | AETERM |
| Data type: | text |
| Current Length: | 100 |
| New Length: | 100 |
| Current Significant Digits: | |
| New Significant Digits: | -1 |
| Current Role: | Topic |
| New Role | Topic |
| Current Role CodeList: | |
| New Role CodeList | CL.MEDDRA - MedDRA Adverse Events Dictionary (text) |
| Current Origin: | NONE DEFINED YET |
| Edit Origin: | Edit |
| Comment: | |
| External document for comment | |
| Current CodeList | NO CODELIST ASSIGNED |
| New CodeList: | Select CodeList |
| Description: | Reported Term for the Adverse Event |
| current def:DisplayFormat: | |
| New def:DisplayFormat: | |
| current ValueList OID: | NO VALUELIST ASSIGNED |
| New ValueList OID | NO VALUELIST |

You can now add the "Origin" by checking the checkbox followed by clicking the "Edit" button[5]. This gives:


Designing/Updating Origin for Item: AETERM

Origin type:
- [✓ Assigned]
- Protocol
- Derived
- Electronic Data Transfer
- CRF

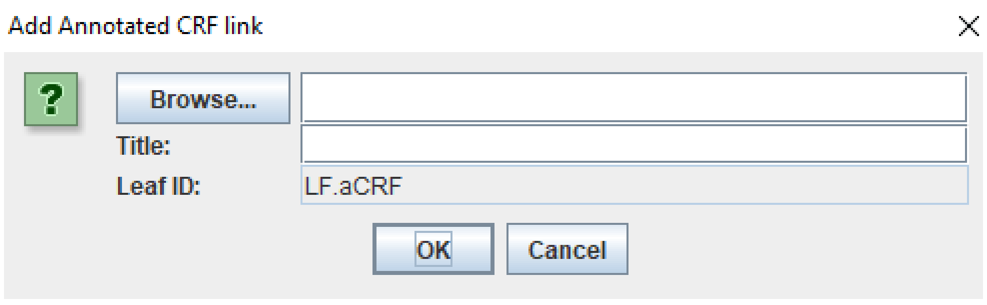
Document (leaf) ID:
No def:leaf elements have been defined yet
- No page details
- Page list (physical reference)
- Named destinations

Page list / List of named destinations
- Page range: first page - last page

First page:
Last page:

In case you did not assign an "annotated CRF", you are notified and you will need to first assign an "annotated CRF" using the menu "Insert - Link to Annotated CRF".

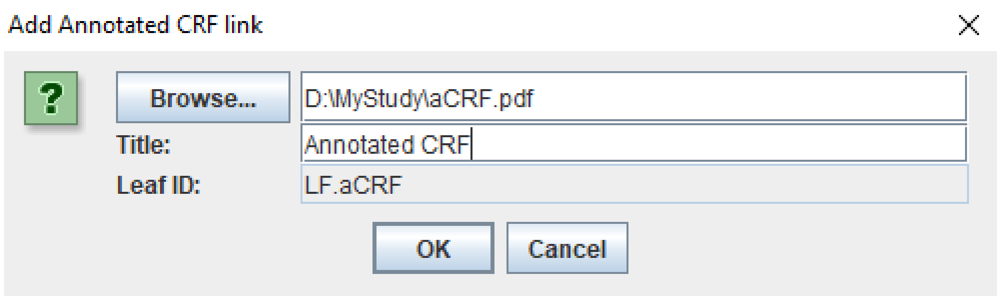
[5]In case of Define-XML 2.1, you will get a slightly different dialog.


Add Annotated CRF link

Title:

Leaf ID: LF.aCRF

You can use the "Browse" button to provide the link to the annotated CRF, and you should provide a title. For example:

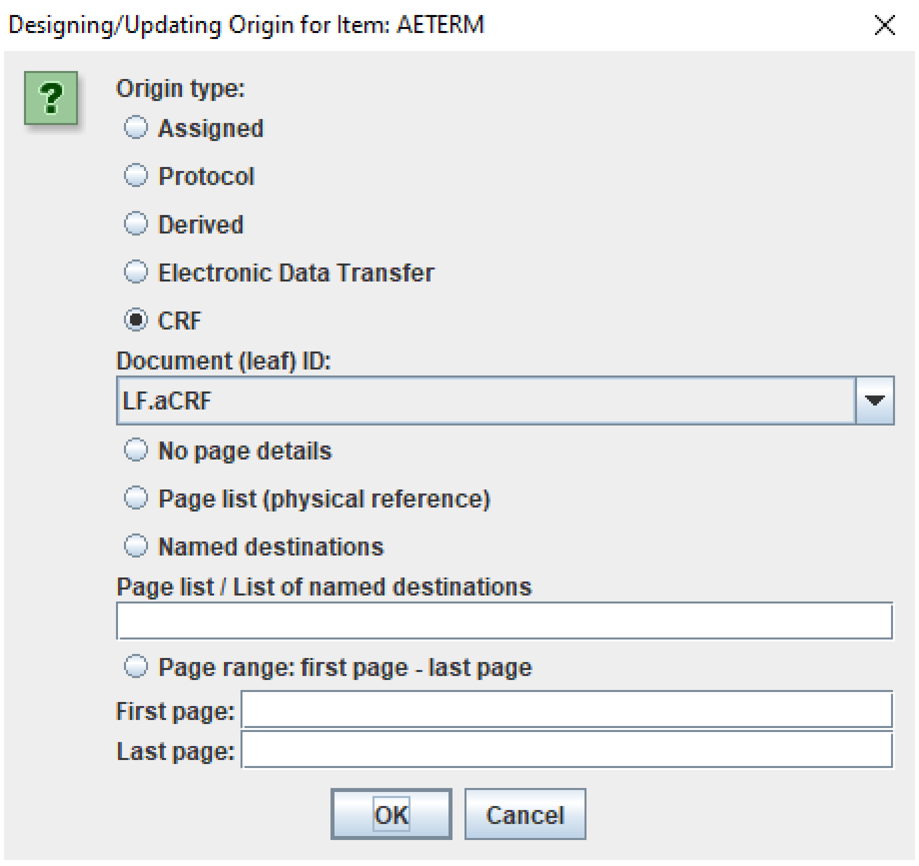

Add Annotated CRF link

D:\MyStudy\aCRF.pdf

Title: Annotated CRF

Leaf ID: LF.aCRF

We can then go back to "Edit - SDTM Variable Properties", and use "Edit Origin"
In this case, it is clear that the origin is „CRF", so we need to check that checkbox:


Designing/Updating Origin for Item: AETERM

Origin type:
- ( ) Assigned
- ( ) Protocol
- ( ) Derived
- ( ) Electronic Data Transfer
- [✓ CRF]

Document (leaf) ID: LF.aCRF

- ( ) No page details
- ( ) Page list (physical reference)
- ( ) Named destinations

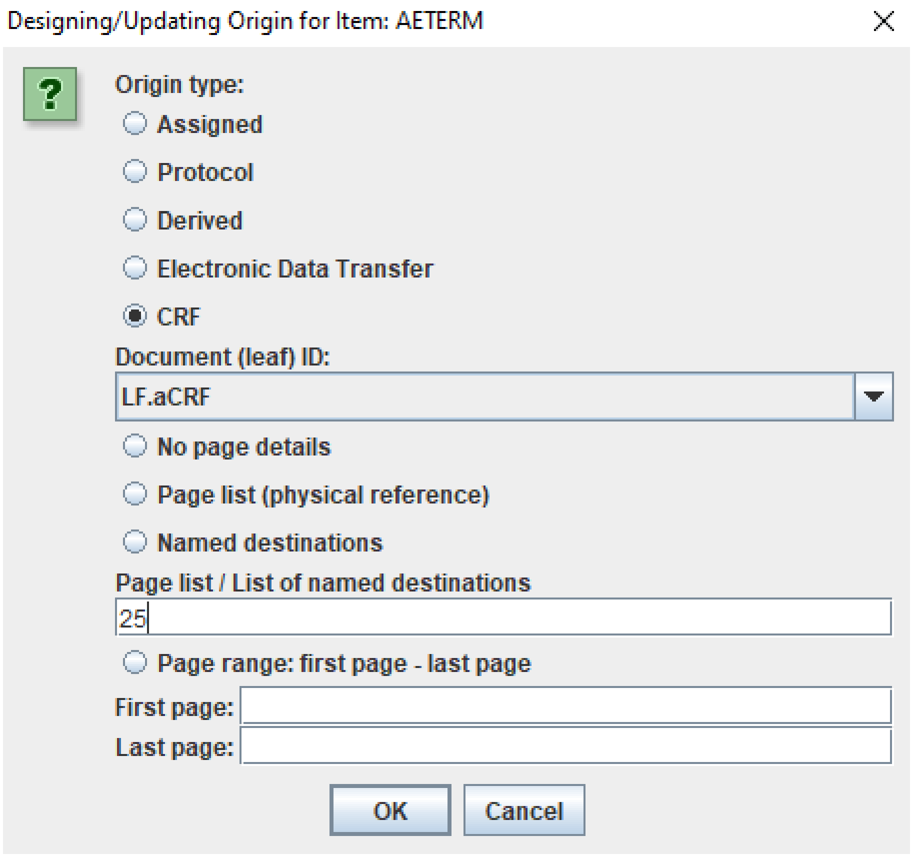
Page list / List of named destinations:

- ( ) Page range: first page - last page

First page:

Last page:

and also add the page number (or a list of blank-separated values) to the field "Page list":


Designing/Updating Origin for Item: AETERM

Origin type:

- ( ) Assigned
- ( ) Protocol
- ( ) Derived
- ( ) Electronic Data Transfer
- (●) CRF

Document (leaf) ID: LF.aCRF

- ( ) No page details
- ( ) Page list (physical reference)
- ( ) Named destinations

Page list / List of named destinations: 25

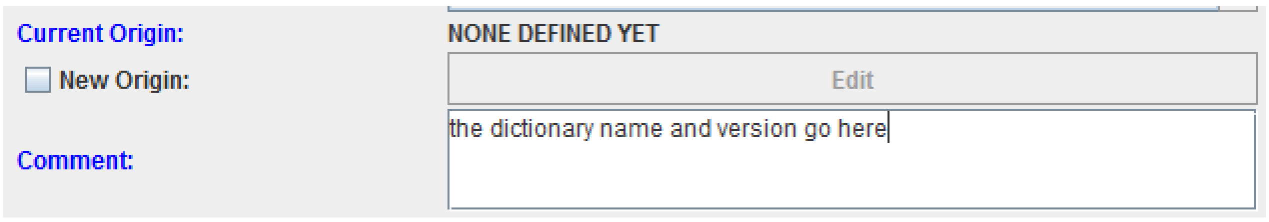
- ( ) Page range: first page - last page

First page:

Last page:

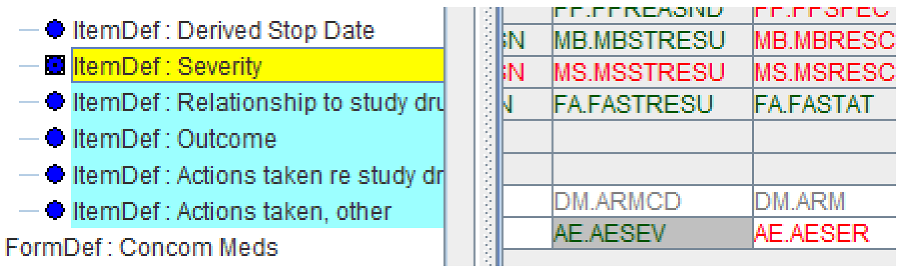
If you do not have an annotated CRF in PDF format yet, you can of course skip this step, and add the information later.

The second SDTM variable we want to develop a mapping for is AE.AEDECOD ("Dictionary-derived term", usually the MedDRA preferred term). As we haven't got anything better, we just use the same mapping as for AE.AETERM for the moment[6].
The "CDISC Notes" (use CTRL-H to see them) tell us that the sponsor need to give the dictionary name and version in the "Comments" column in the define.xml document. This can easily be done by selecting the AE.AEDECOD cell, and then use the menu "Edit -> SDTM Variable Properties". We can then add the information in the input field for "Comment":


| | |
|---|---|
| Current Origin: | NONE DEFINED YET |
| ☐ New Origin: | Edit |
| Comment: | the dictionary name and version go here |

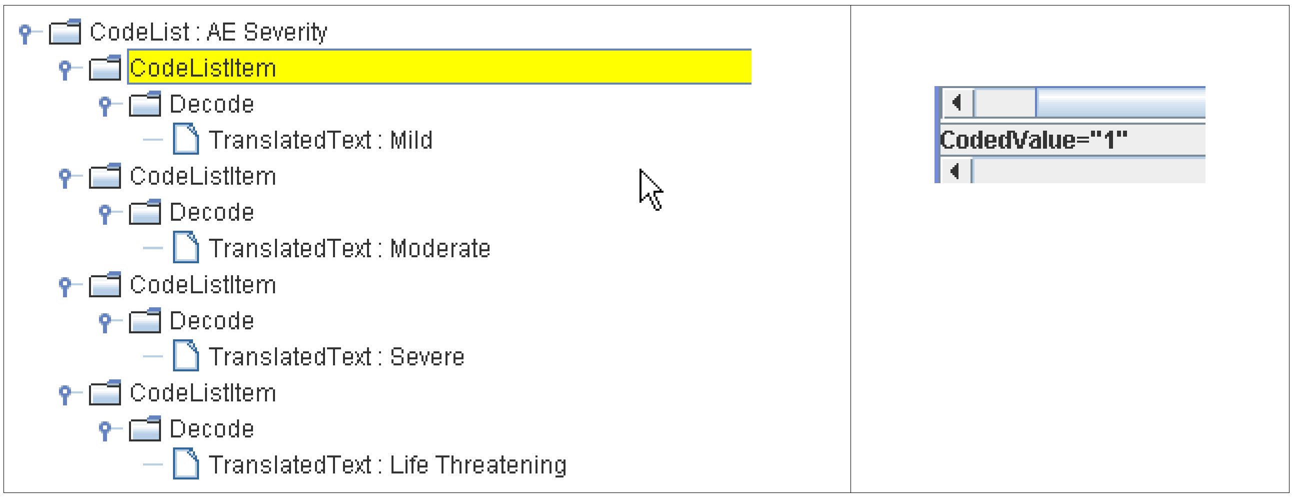
The next SDTM variable we can develop a mapping for is the variable AE.AESEV (Severity/Intensity). When we select the cell, we see that there is an ODM "hot candidate" IT.AESEV ("Severity"), which is however governed by a codelist.

[6]If you have a MedDRA database or tool, we can help you generating a connection to it for mapping purposes.


ItemDef : Derived Stop Date
ItemDef : Severity
ItemDef : Relationship to study dru
ItemDef : Outcome
ItemDef : Actions taken re study dr
ItemDef : Actions taken, other
FormDef : Concom Meds

| | | |
|---|---|---|
| N | MB.MBSTRESU | MB.MBRESC |
| N | MS.MSSTRESU | MS.MSRESC |
| N | FA.FASTRESU | FA.FASTAT |
| | | |
| | | |
| | DM.ARMCD | DM.ARM |
| | AE.AESEV | AE.AESER |

As indicated by the square around the "traffic light" and the blue color of it (indicating there is an associated codelist). To see the details of the codelist at the ODM side, select the Item "Severity" and then use the menu "Navigate -> Select associated CodeList" (Ctrl-F2). The selection jumps to the associated CodeList in the ODM tree and expands its tree node:


CodeList : AE Severity
CodeListItem
Decode
TranslatedText : Mild
CodeListItem
Decode
TranslatedText : Moderate
CodeListItem
Decode
TranslatedText : Severe
CodeListItem
Decode
TranslatedText : Life Threatening

CodedValue="1"

We observe that there are four possibilities with coded values "1", "2", "3" or "4" (these can be seen in the status field at the bottom when selecting a "CodeListItem", and decoded values "Mild", "Moderate", "Severe" and "Life Threatening".

The SDTM-IG tells us that there is controlled terminology for AESEV and provides a reference to the NCI website. There we can find that there is a codelist „CL.C66769.AESEV" which however only has 3 allowed values: „MILD", „MODERATE" and „SEVERE". It also states that the codelist is non-extensible. Unfortunately the SDTM-IG is also not very clear on whether one <u>must</u> use that codelist. It states: "The severity or intensity of the event. <u>Examples</u>: MILD, MODERATE, SEVERE". Especially, the wording "Examples" is confusing.

We now have the choice between two approaches:

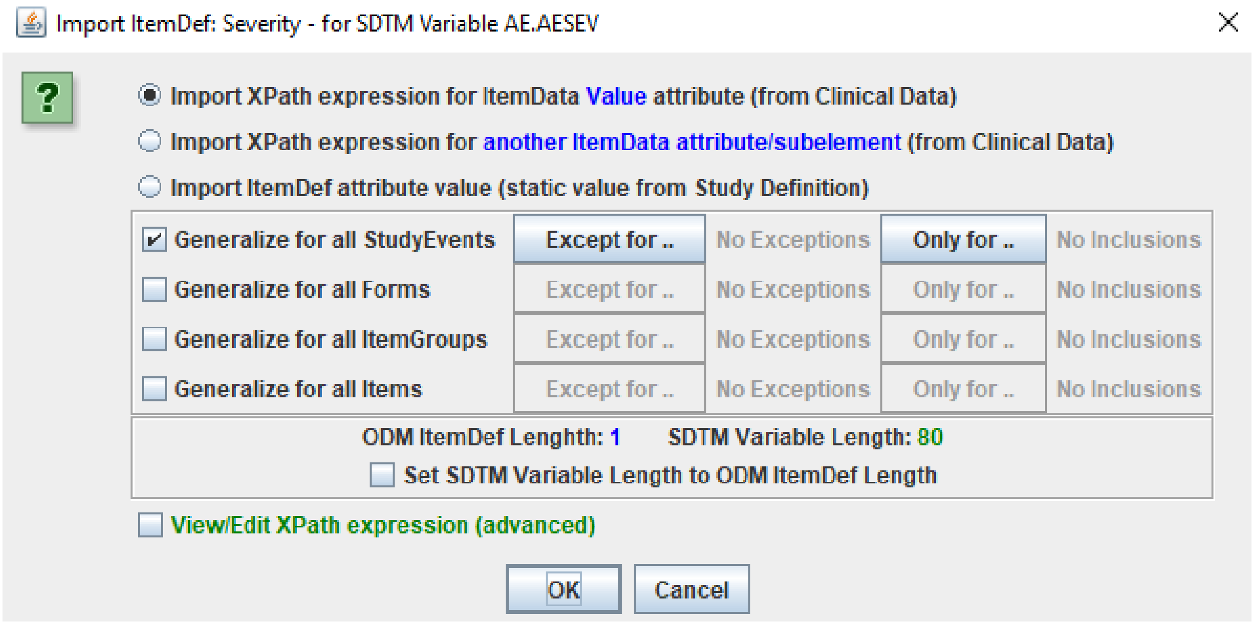
a) map the 4 possibible values from the study design to the three values of the "recommended" SDTM-IG codelist with only 3 values. If we do so, we should document this in the "reviewers guide" but also in the "annotated CRF"

b) copy our own "SEVERITY" codelist from the ODM (study design) to the SDTM, and use that.

We will use the first approach (just as an example[7]).

Now just drag-and-drop the ItemDef "Severity" to the SDTM cell "AE.AESEV". The following dialog shows up:

[7]In such cases, the mapper should decide himself / herself which of the 2 approaches is most appropriate


Import ItemDef: Severity - for SDTM Variable AE.AESEV

[✓ Import XPath expression for ItemData Value attribute (from Clinical Data)]
Import XPath expression for another ItemData attribute/subelement (from Clinical Data)
Import ItemDef attribute value (static value from Study Definition)

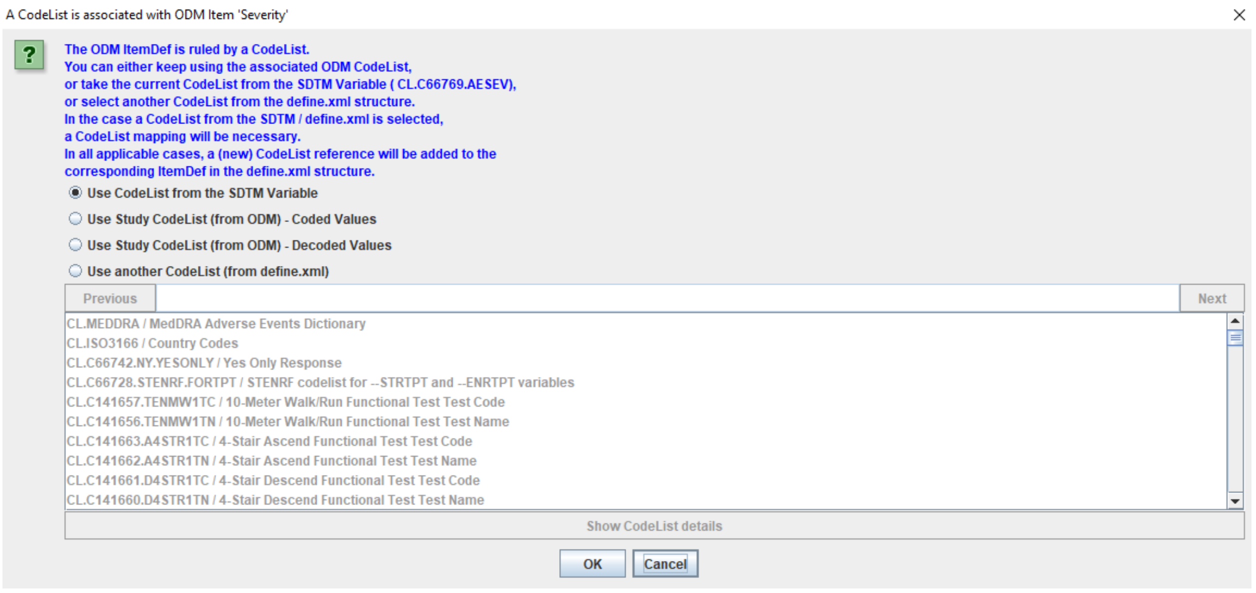
|  |  |  |  |  |
|---|---|---|---|---|
| ✓ Generalize for all StudyEvents | Except for .. | No Exceptions | Only for .. | No Inclusions |
| Generalize for all Forms | Except for .. | No Exceptions | Only for .. | No Inclusions |
| Generalize for all ItemGroups | Except for .. | No Exceptions | Only for .. | No Inclusions |
| Generalize for all Items | Except for .. | No Exceptions | Only for .. | No Inclusions |

ODM ItemDef Lenghth: 1  SDTM Variable Length: 80
Set SDTM Variable Length to ODM ItemDef Length

View/Edit XPath expression (advanced)

OK Cancel

This time, we do NOT check the button "Set SDTM Variable Length to ODM ItemDef Length" as we will use a different codelist on the SDTM side. Do not forget to check the box "Generalize for all Study Events" if it was not automatically checked yet. Clicking "OK" leads to:


A CodeList is associated with ODM Item 'Severity'

The ODM ItemDef is ruled by a CodeList.
You can either keep using the associated ODM CodeList,
or take the current CodeList from the SDTM Variable ( CL.C66769.AESEV),
or select another CodeList from the define.xml structure.
In the case a CodeList from the SDTM / define.xml is selected,
a CodeList mapping will be necessary.
In all applicable cases, a (new) CodeList reference will be added to the
corresponding ItemDef in the define.xml structure.

[✓ Use CodeList from the SDTM Variable]
Use Study CodeList (from ODM) - Coded Values
Use Study CodeList (from ODM) - Decoded Values
Use another CodeList (from define.xml)

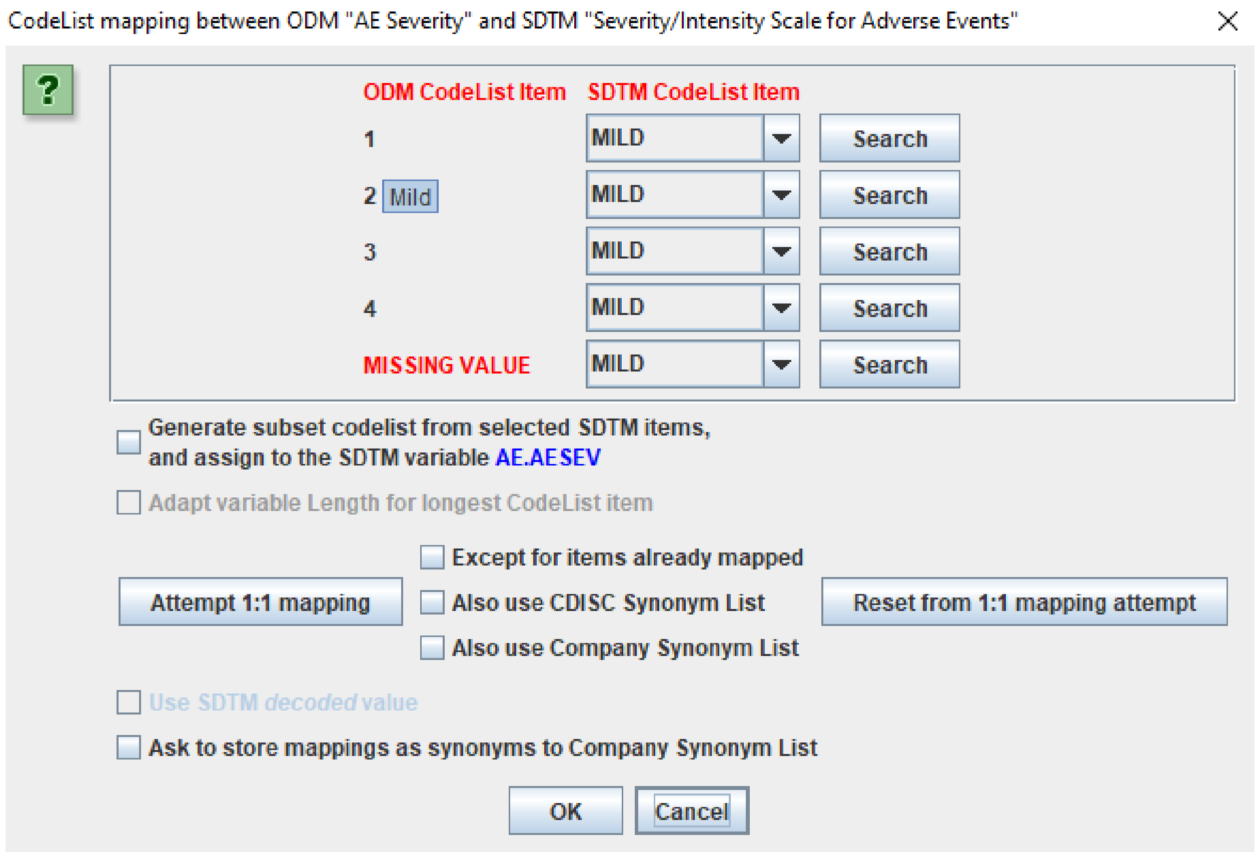
Previous Next

CL.MEDDRA / MedDRA Adverse Events Dictionary
CL.ISO3166 / Country Codes
CL.C66742.NY.YESONLY / Yes Only Response
CL.C66728.STENRF.FORTPT / STENRF codelist for --STRTPT and --ENRTPT variables
CL.C141657.TENMW1TC / 10-Meter Walk/Run Functional Test Test Code
CL.C141656.TENMW1TN / 10-Meter Walk/Run Functional Test Test Name
CL.C141663.A4STR1TC / 4-Stair Ascend Functional Test Test Code
CL.C141662.A4STR1TN / 4-Stair Ascend Functional Test Test Name
CL.C141661.D4STR1TC / 4-Stair Descend Functional Test Test Code
CL.C141660.D4STR1TN / 4-Stair Descend Functional Test Test Name

Show CodeList details

OK Cancel

Where we select „Use CodeList from SDTM variable" (which is the CDISC-NCI codelist). Then clicking "OK" leads to:

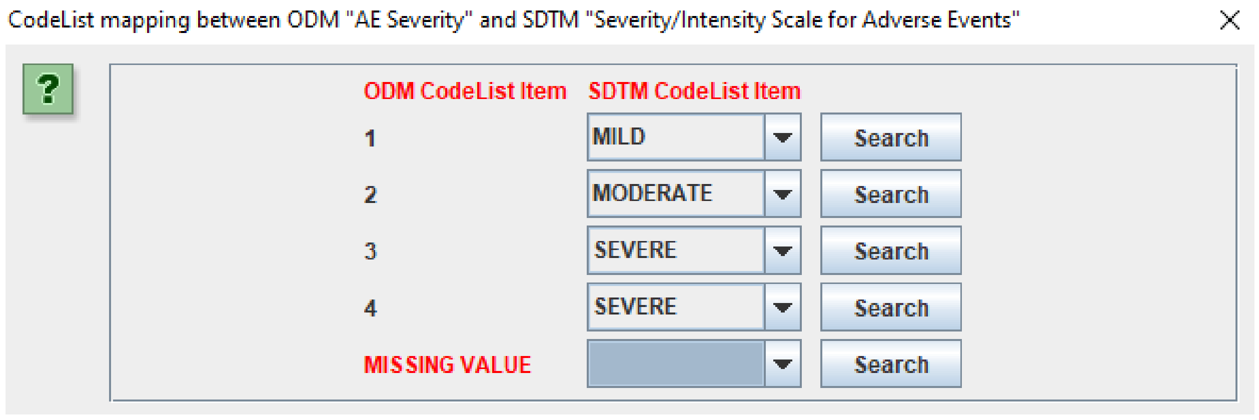


CodeList mapping between ODM "AE Severity" and SDTM "Severity/Intensity Scale for Adverse Events"

| ODM CodeList Item | SDTM CodeList Item | |
|---|---|---|
| 1 | MILD | Search |
| 2 Mild | MILD | Search |
| 3 | MILD | Search |
| 4 | MILD | Search |
| MISSING VALUE | MILD | Search |

☐ Generate subset codelist from selected SDTM items, and assign to the SDTM variable AE.AESEV
☐ Adapt variable Length for longest CodeList item
Attempt 1:1 mapping
☐ Except for items already mapped
☐ Also use CDISC Synonym List
☐ Also use Company Synonym List
Reset from 1:1 mapping attempt
☐ Use SDTM *decoded* value
☐ Ask to store mappings as synonyms to Company Synonym List
OK Cancel

which allows us to do the mapping between both the codelists.
As we know that "1" stands for "Mild", "2" for "Moderate" and "3" for "Severe", we just use the dropdowns leading to:

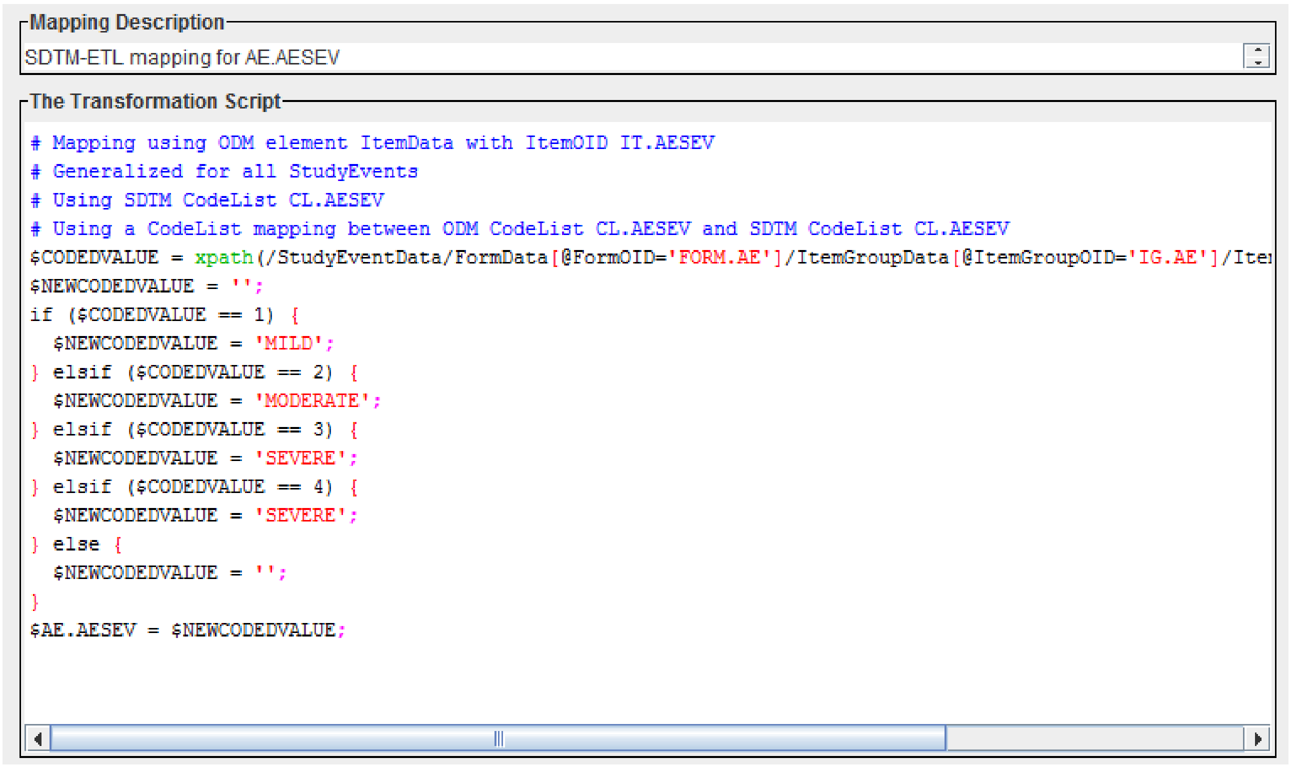


CodeList mapping between ODM "AE Severity" and SDTM "Severity/Intensity Scale for Adverse Events"

| ODM CodeList Item | SDTM CodeList Item | |
|---|---|---|
| 1 | MILD | Search |
| 2 | MODERATE | Search |
| 3 | SEVERE | Search |
| 4 | SEVERE | Search |
| MISSING VALUE | | Search |

where we map "life threatening" to "severe".
For "Missing" value, we just map to "blank" - this will give a null (empty) value in the later result.
So we have:

The button "Attempt 1:1 mapping based on coded values" can be used in case of long codelists. The system will then try to compare both (based on word similarity as well for the coded as the decoded value) and come with a suitable proposal for the mappings. After clicking "OK", the "mapping editor" shows up and displays the result:


Mapping Description

SDTM-ETL mapping for AE.AESEV

The Transformation Script

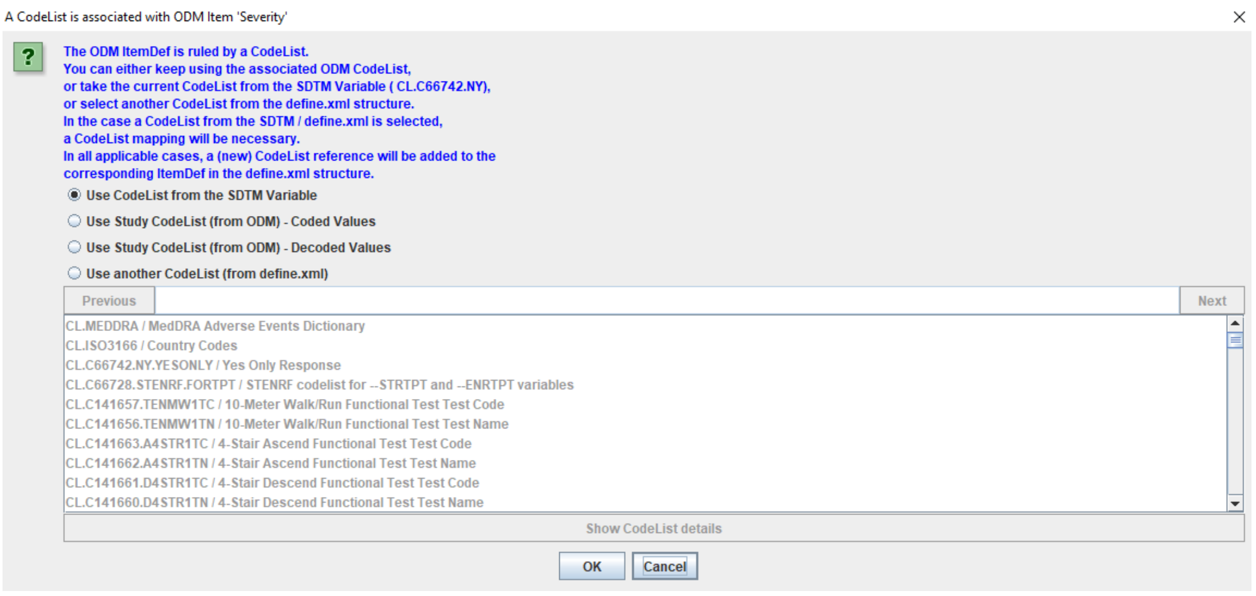
```
# Mapping using ODM element ItemData with ItemOID IT.AESEV
# Generalized for all StudyEvents
# Using SDTM CodeList CL.AESEV
# Using a CodeList mapping between ODM CodeList CL.AESEV and SDTM CodeList CL.AESEV
$CODEDVALUE = xpath(/StudyEventData/FormData[@FormOID='FORM.AE']/ItemGroupData[@ItemGroupOID='IG.AE']/Ite
$NEWCODEDVALUE = '';
if ($CODEDVALUE == 1) {
  $NEWCODEDVALUE = 'MILD';
} elsif ($CODEDVALUE == 2) {
  $NEWCODEDVALUE = 'MODERATE';
} elsif ($CODEDVALUE == 3) {
  $NEWCODEDVALUE = 'SEVERE';
} elsif ($CODEDVALUE == 4) {
  $NEWCODEDVALUE = 'SEVERE';
} else {
  $NEWCODEDVALUE = '';
}
$AE.AESEV = $NEWCODEDVALUE;
```

So the system has automatically generated a mapping script – no need for typing!
Of course one can always edit/adapt the mapping script, but in this case, this is not necessary at all.

Also remark that comment lines (all starting with „#" have automatically been generated).

The next SDTM variable on our list is AE.AESER ("Is this a serious event?"). The possible values are "Y" (Yes) and "N" (No). We see that there is no corresponding item in the ODM, so we will (arbitrarily) add "N" values for mild and moderate adverse events, and "Y" for severe and life threatening adverse events.

Drag-and-drop from IT.AESEV (ItemDef: Severity)  to the SDTM cell AE.AESER. Check again to generalize for all visits (StudyEvents). We get:

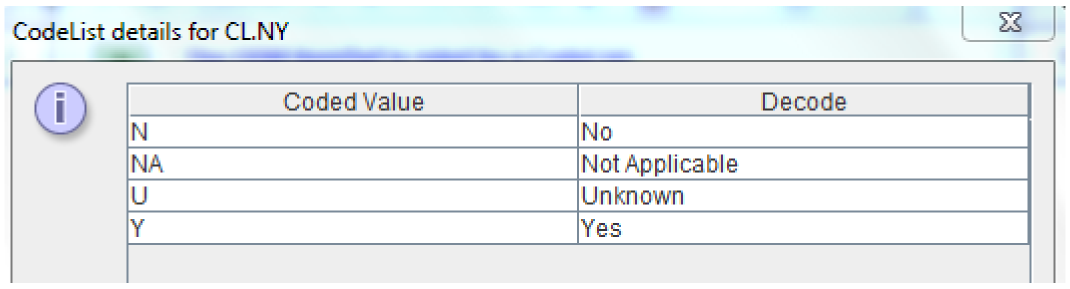

A CodeList is associated with ODM Item 'Severity'

The ODM ItemDef is ruled by a CodeList.
You can either keep using the associated ODM CodeList,
or take the current CodeList from the SDTM Variable ( CL.C66742.NY),
or select another CodeList from the define.xml structure.
In the case a CodeList from the SDTM / define.xml is selected,
a CodeList mapping will be necessary.
In all applicable cases, a (new) CodeList reference will be added to the
corresponding ItemDef in the define.xml structure.

[✓ Use CodeList from the SDTM Variable]
Use Study CodeList (from ODM) - Coded Values
Use Study CodeList (from ODM) - Decoded Values
Use another CodeList (from define.xml)

CL.MEDDRA / MedDRA Adverse Events Dictionary
CL.ISO3166 / Country Codes
CL.C66742.NY.YESONLY / Yes Only Response
CL.C66728.STENRF.FORTPT / STENRF codelist for --STRTPT and --ENRTPT variables
CL.C141657.TENMW1TC / 10-Meter Walk/Run Functional Test Test Code
CL.C141656.TENMW1TN / 10-Meter Walk/Run Functional Test Test Name
CL.C141663.A4STR1TC / 4-Stair Ascend Functional Test Test Code
CL.C141662.A4STR1TN / 4-Stair Ascend Functional Test Test Name
CL.C141661.D4STR1TC / 4-Stair Descend Functional Test Test Code
CL.C141660.D4STR1TN / 4-Stair Descend Functional Test Test Name

We better choose the codelist from the SDTM Variable. Check the radiobutton and then click the button "Show Codelist Details":

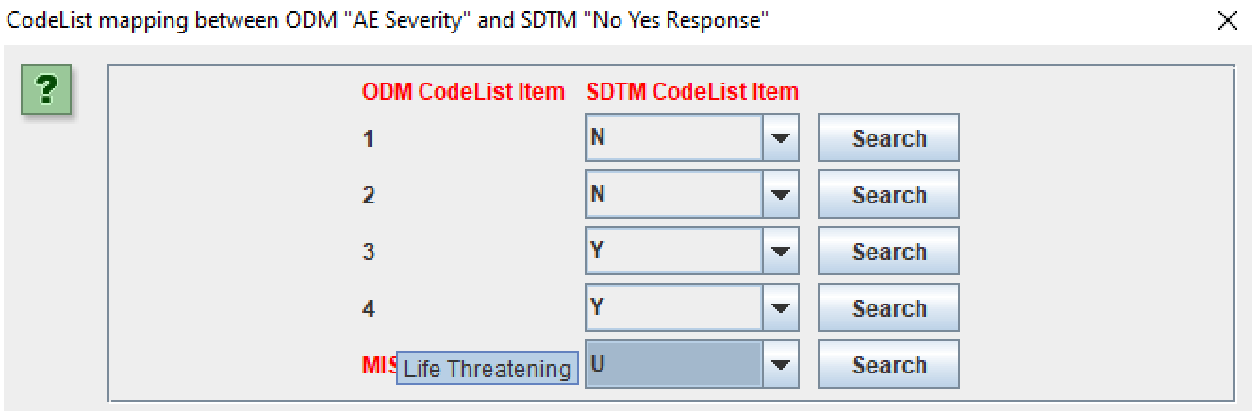


CodeList details for CL.NY

| Coded Value | Decode |
|---|---|
| N | No |
| NA | Not Applicable |
| U | Unknown |
| Y | Yes |

Remark that dependent on which version of CDISC-CT you have chosen when loading the template, you might see slightly different things here.

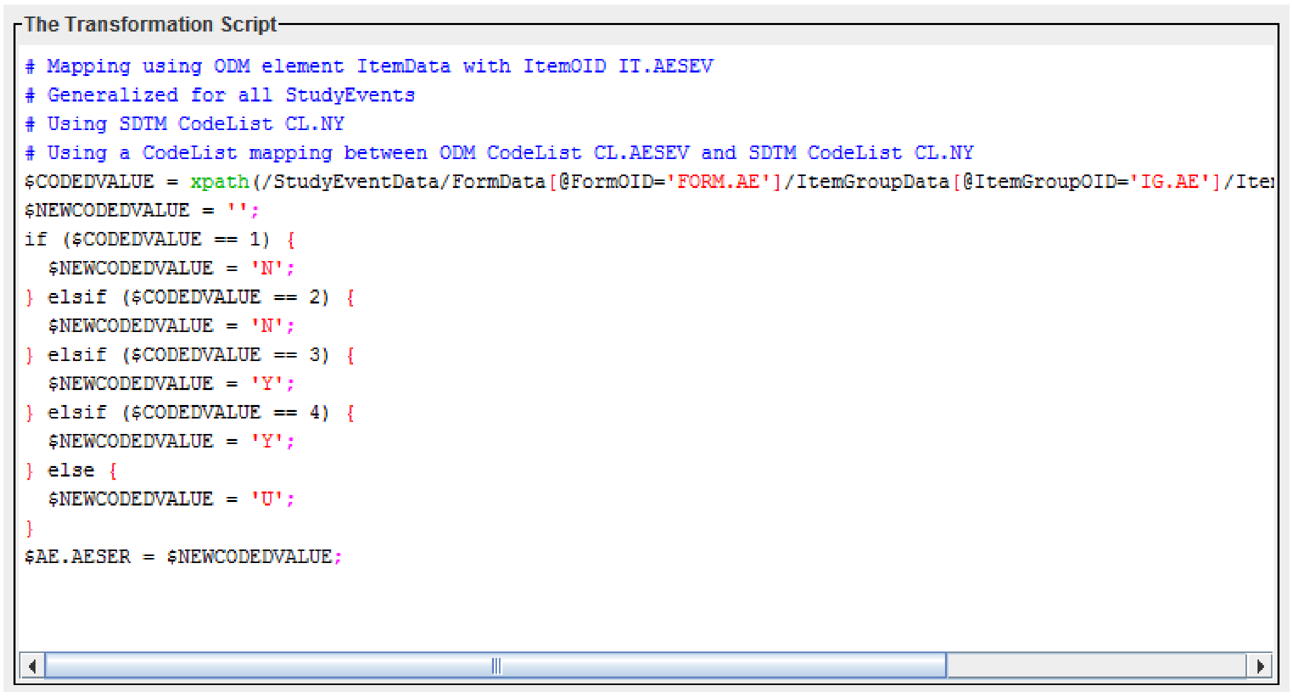
Again, the codelist mapping tool is then displayed. We map to:



CodeList mapping between ODM "AE Severity" and SDTM "No Yes Response"

| ODM CodeList Item | SDTM CodeList Item |
|---|---|
| 1 | N |
| 2 | N |
| 3 | Y |
| 4 | Y |
| MIS Life Threatening | U |

Which leads to the automatically created mapping script:

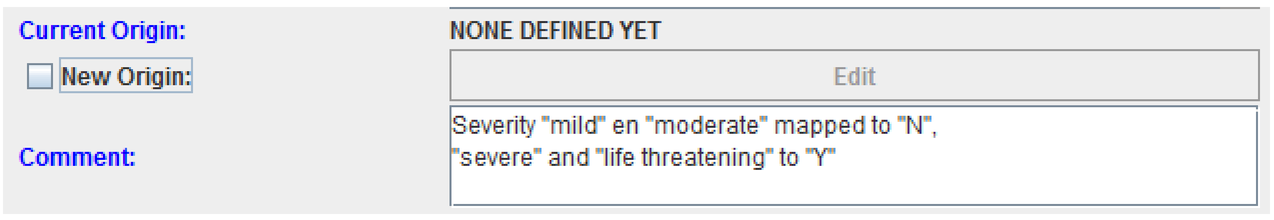


The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.AESEV
# Generalized for all StudyEvents
# Using SDTM CodeList CL.NY
# Using a CodeList mapping between ODM CodeList CL.AESEV and SDTM CodeList CL.NY
$CODEDVALUE = xpath(/StudyEventData/FormData[@FormOID='FORM.AE']/ItemGroupData[@ItemGroupOID='IG.AE']/Ite
$NEWCODEDVALUE = '';
if ($CODEDVALUE == 1) {
  $NEWCODEDVALUE = 'N';
} elsif ($CODEDVALUE == 2) {
  $NEWCODEDVALUE = 'N';
} elsif ($CODEDVALUE == 3) {
  $NEWCODEDVALUE = 'Y';
} elsif ($CODEDVALUE == 4) {
  $NEWCODEDVALUE = 'Y';
} else {
  $NEWCODEDVALUE = 'U';
}
$AE.AESER = $NEWCODEDVALUE;
```

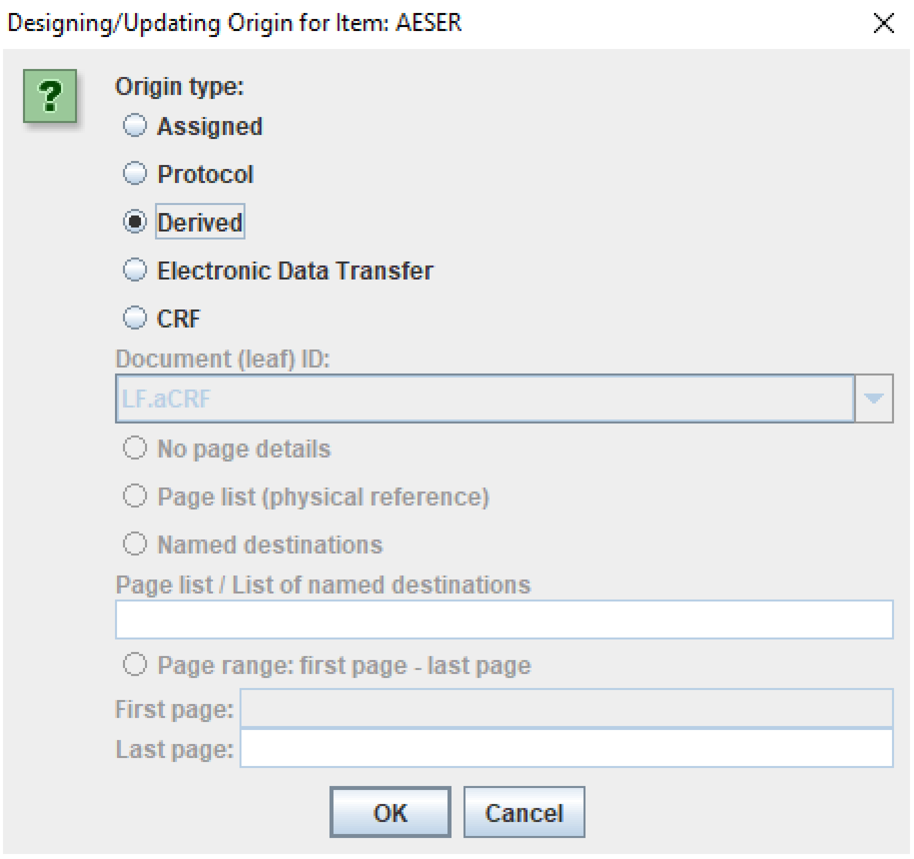
Also remark the automatically created comment line, allowing us later to remember how the

mapping script was generated[8].
In case such an arbitrary decision is made, it is surely not a bad habit to explain that decision in the "Comments" field of the define.xml. You can do so using the menu "Edit => SDTM Variable Properties", and add your comment in the field "Comment":



| | |
|---|---|
| Current Origin: | NONE DEFINED YET |
| ☐ New Origin: | Edit |
| Comment: | Severity "mild" en "moderate" mapped to "N", "severe" and "life threatening" to "Y" |

For a regulatory submission, define.xml 2.0 also requires that the "Origin" is provided, either on the variable level or on the valuelist level. We can easily add it using the "New Origin" checkbox followed by "Edit":



Designing/Updating Origin for Item: AESER

Origin type:
- Assigned
- Protocol
- [✓ Derived]
- Electronic Data Transfer
- CRF

Document (leaf) ID: LF.aCRF

- No page details
- Page list (physical reference)
- Named destinations

Page list / List of named destinations

- Page range: first page - last page

First page:
Last page:

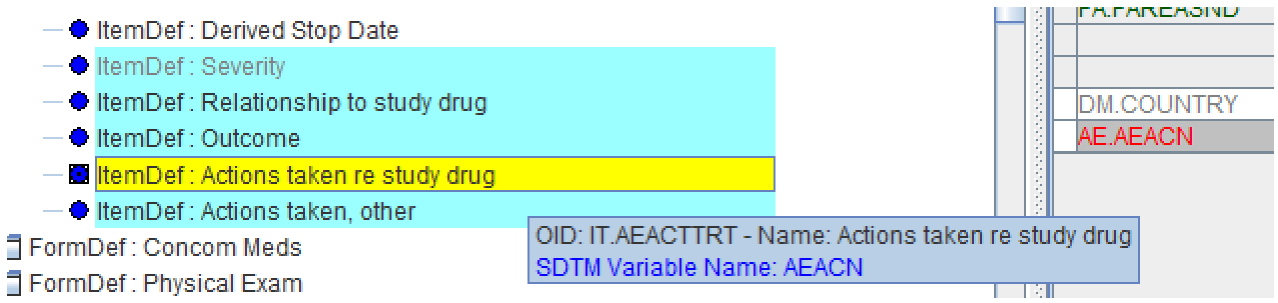
OK Cancel

and in this case set it to "Derived"[9].

Note: when having finished mapping a domain / dataset, it is surely a good idea to check whether each variable has been assigned an "Origin". The absence of it is NOT marked as an error even when using Pinnacle21[10], as the latter cannot know whether the context of the define.xml is a regulatory submission.

[8]And which can also be read by the FDA reviewer when inspecting the "Methods" in the submitted define.xml file.
[9] In the case of Define-XML 2.1, you will see a slightly different dialog.
[10] This will be possible using the long expected CORE (CDISC Open Rules Engine), in combination with Define-XML 2.1

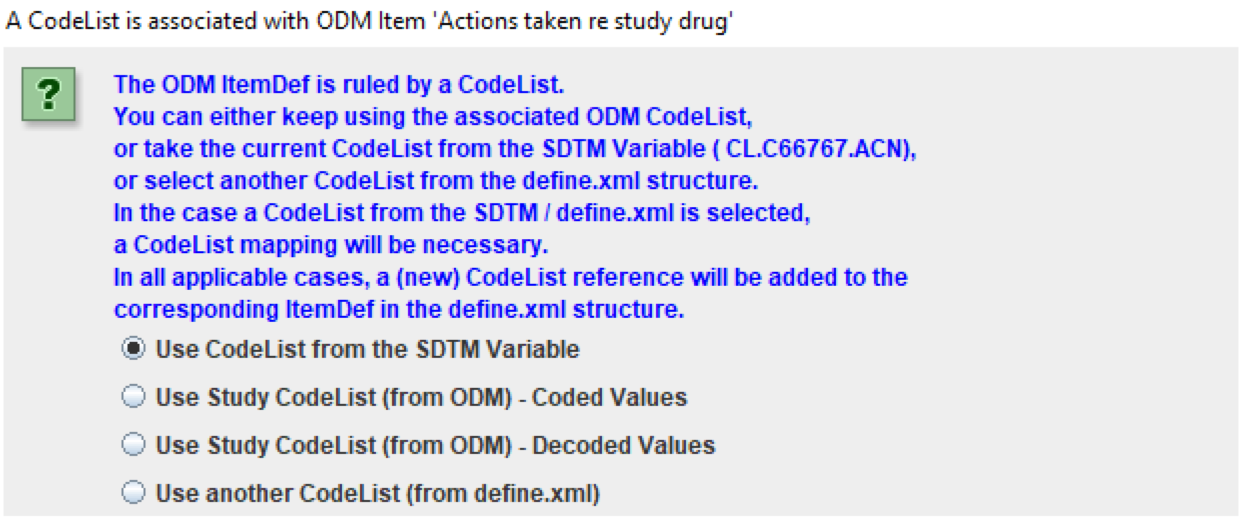
The next SDTM variable is AE.AEACN (Action taken with study treatment).
When clicking the cell, we see that there is an ODM "hot candidate" "Actions take re study drug". Alternatively, we can search for a "hot candidate" on the ODM side using the menu "Navigate – Find hot SDTM Candidate". We find:



ItemDef : Derived Stop Date
ItemDef : Severity
ItemDef : Relationship to study drug
ItemDef : Outcome
ItemDef : Actions taken re study drug
ItemDef : Actions taken, other
FormDef : Concom Meds
FormDef : Physical Exam
DM.COUNTRY
AE.AEACN

OID: IT.AEACTTRT - Name: Actions taken re study drug
SDTM Variable Name: AEACN

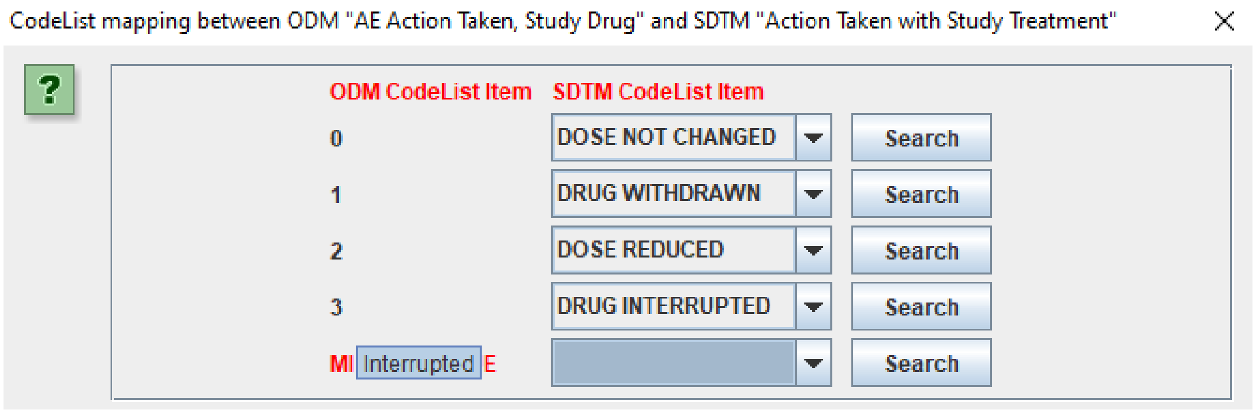
Drag-and-drop, and generalizing for all StudyEvents gives us the dialog asking us which codelist to use. Again we can use the already associated SDTM codelist:


A CodeList is associated with ODM Item 'Actions taken re study drug'

The ODM ItemDef is ruled by a CodeList.
You can either keep using the associated ODM CodeList,
or take the current CodeList from the SDTM Variable ( CL.C66767.ACN),
or select another CodeList from the define.xml structure.
In the case a CodeList from the SDTM / define.xml is selected,
a CodeList mapping will be necessary.
In all applicable cases, a (new) CodeList reference will be added to the
corresponding ItemDef in the define.xml structure.

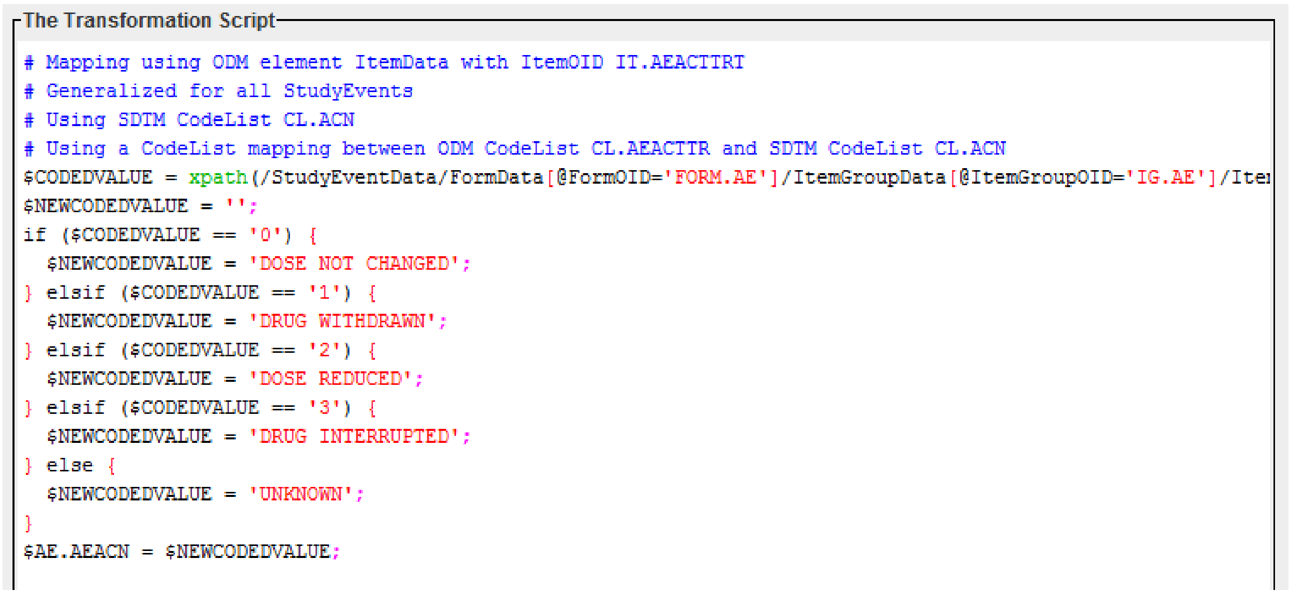
- [✓ Use CodeList from the SDTM Variable]
- Use Study CodeList (from ODM) - Coded Values
- Use Study CodeList (from ODM) - Decoded Values
- Use another CodeList (from define.xml)

Clicking "OK" gives us the codelist mapping wizard, which easily leads us to the following mapping:


CodeList mapping between ODM "AE Action Taken, Study Drug" and SDTM "Action Taken with Study Treatment"

| ODM CodeList Item | SDTM CodeList Item |
|---|---|
| 0 | DOSE NOT CHANGED |
| 1 | DRUG WITHDRAWN |
| 2 | DOSE REDUCED |
| 3 | DRUG INTERRUPTED |
| MI Interrupted E | |

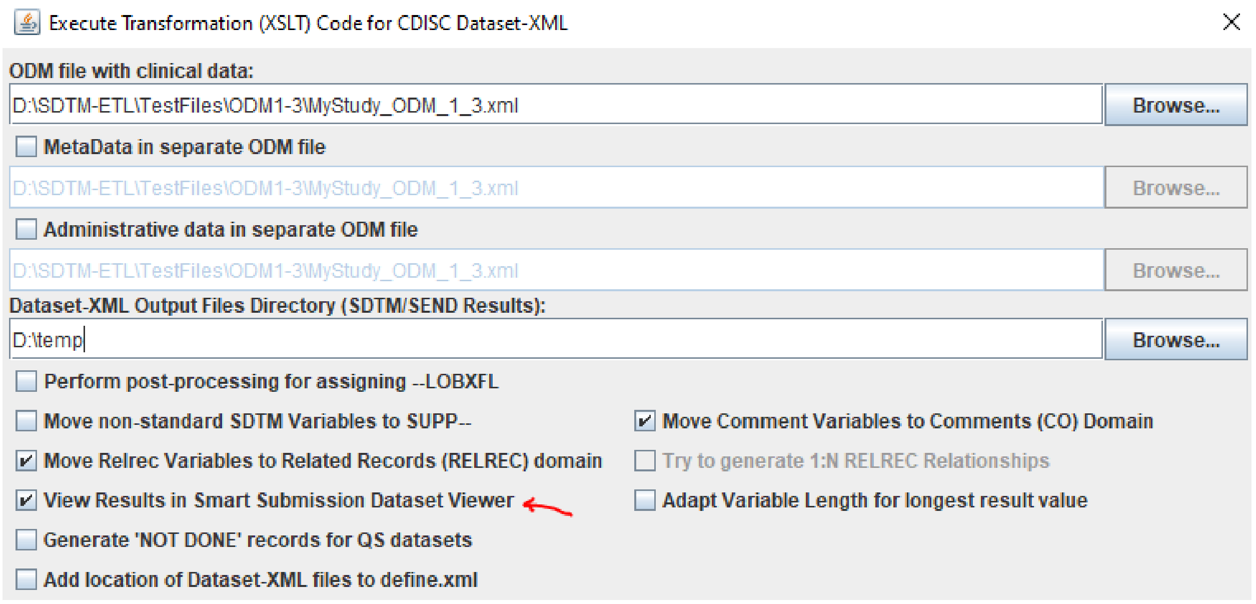
and again, the mapping script is automatically generated:



The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.AEACTTRT
# Generalized for all StudyEvents
# Using SDTM CodeList CL.ACN
# Using a CodeList mapping between ODM CodeList CL.AEACTTR and SDTM CodeList CL.ACN
$CODEDVALUE = xpath(/StudyEventData/FormData[@FormOID='FORM.AE']/ItemGroupData[@ItemGroupOID='IG.AE']/Ite
$NEWCODEDVALUE = '';
if ($CODEDVALUE == '0') {
  $NEWCODEDVALUE = 'DOSE NOT CHANGED';
} elsif ($CODEDVALUE == '1') {
  $NEWCODEDVALUE = 'DRUG WITHDRAWN';
} elsif ($CODEDVALUE == '2') {
  $NEWCODEDVALUE = 'DOSE REDUCED';
} elsif ($CODEDVALUE == '3') {
  $NEWCODEDVALUE = 'DRUG INTERRUPTED';
} else {
  $NEWCODEDVALUE = 'UNKNOWN';
}
$AE.AEACN = $NEWCODEDVALUE;
```

At this point, let us try to execute the full mapping that we have so far on a set of clinical data, just as a check that we are doing the right things. For this, use the menu "Transform - Generate Transformation (XSLT) Code for CDISC Dataset-XML" (alternatively you can also use „Generate Transformation (XSLT) Code for SAS-XPT") followed by "Excecute Transformation (XSLT) Code":



Execute Transformation (XSLT) Code for CDISC Dataset-XML

ODM file with clinical data: D:\SDTM-ETL\TestFiles\ODM1-3\MyStudy_ODM_1_3.xml

MetaData in separate ODM file: [ ] D:\SDTM-ETL\TestFiles\ODM1-3\MyStudy_ODM_1_3.xml

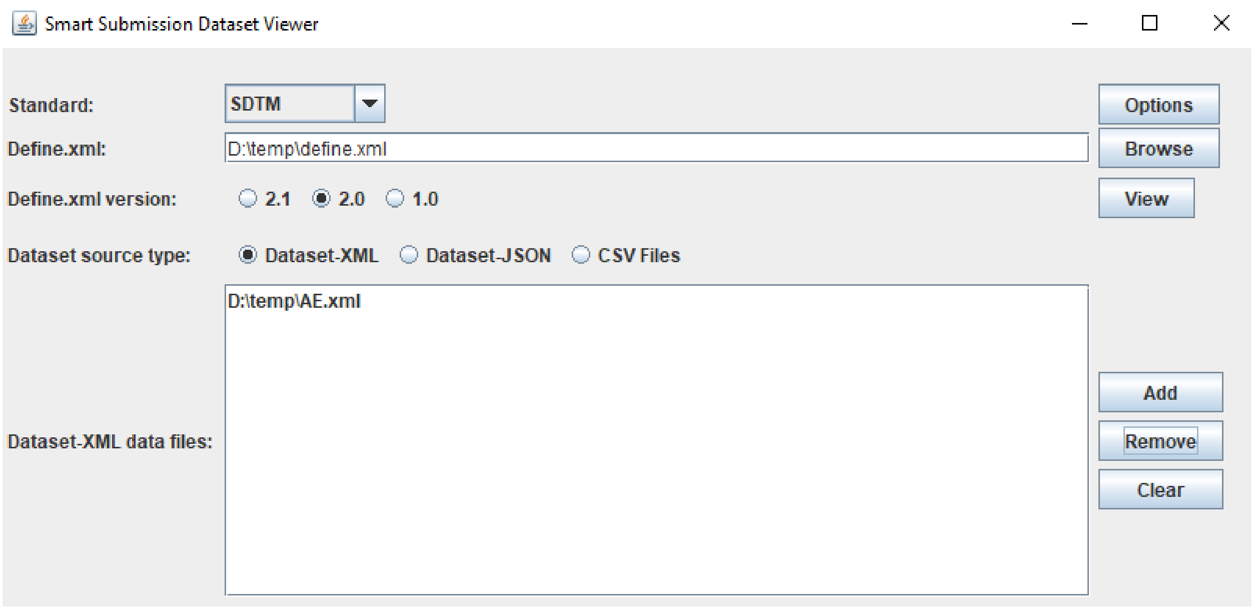
Administrative data in separate ODM file: [ ] D:\SDTM-ETL\TestFiles\ODM1-3\MyStudy_ODM_1_3.xml

Dataset-XML Output Files Directory (SDTM/SEND Results): D:\temp

- [ ] Perform post-processing for assigning --LOBXFL
- [ ] Move non-standard SDTM Variables to SUPP--
- [✓] Move Comment Variables to Comments (CO) Domain
- [✓] Move Relrec Variables to Related Records (RELREC) domain
- [ ] Try to generate 1:N RELREC Relationships
- [✓] View Results in Smart Submission Dataset Viewer
- [ ] Adapt Variable Length for longest result value
- [ ] Generate 'NOT DONE' records for QS datasets
- [ ] Add location of Dataset-XML files to define.xml

Note that there is no checkbox „Split records > 200 characters to SUPP-- records" as this is not necessary anymore when using Dataset-XML instead of SAS-XPT. We select a file with clinical data. As we want to inspect the results, we check the checkbox "View Results in Smart Submission Dataset Viewer".

Then click "Execute Transformation on Clinical Data". The mapping scripts are executed and soon the "Smart Dataset-XML Viewer" shows up:


Smart Submission Dataset Viewer

Standard: SDTM

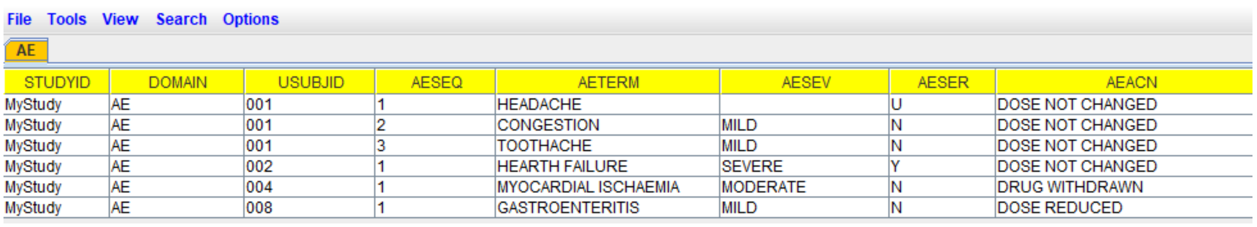
Define.xml: D:\temp\define.xml

Define.xml version: ( ) 2.1 [✓ 2.0] ( ) 1.0

Dataset source type: [✓ Dataset-XML] ( ) Dataset-JSON ( ) CSV Files

Dataset-XML data files: D:\temp\AE.xml

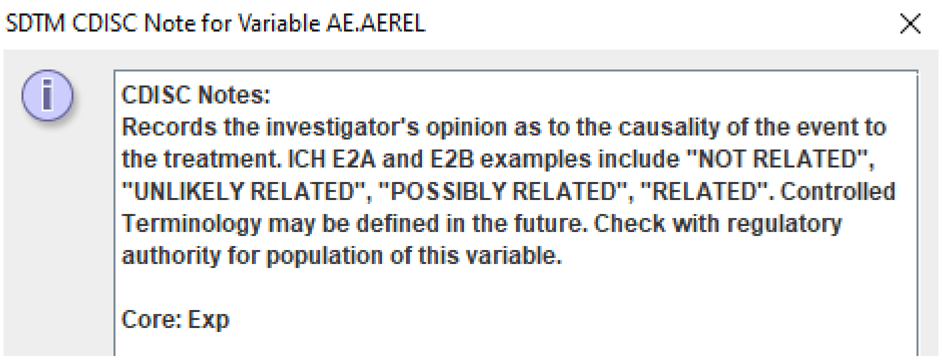
leading to:



| STUDYID | DOMAIN | USUBJID | AESEQ | AETERM | AESEV | AESER | AEACN |
|---|---|---|---|---|---|---|---|
| MyStudy | AE | 001 | 1 | HEADACHE | | U | DOSE NOT CHANGED |
| MyStudy | AE | 001 | 2 | CONGESTION | MILD | N | DOSE NOT CHANGED |
| MyStudy | AE | 001 | 3 | TOOTHACHE | MILD | N | DOSE NOT CHANGED |
| MyStudy | AE | 002 | 1 | HEARTH FAILURE | SEVERE | Y | DOSE NOT CHANGED |
| MyStudy | AE | 004 | 1 | MYOCARDIAL ISCHAEMIA | MODERATE | N | DRUG WITHDRAWN |
| MyStudy | AE | 008 | 1 | GASTROENTERITIS | MILD | N | DOSE REDUCED |

There are a few records only, but the results seem to be OK.

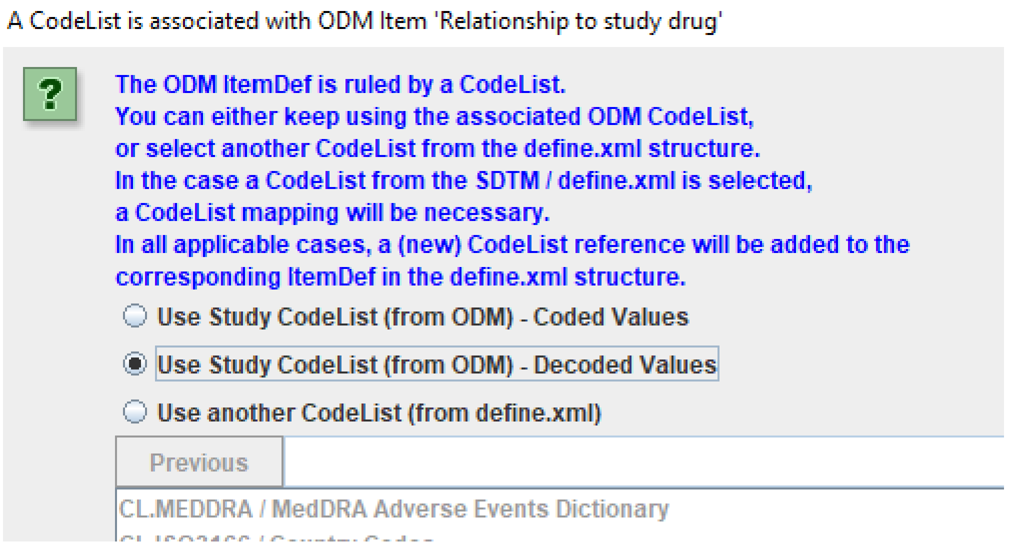
The next SDTM variable we can tackle is AE.AEREL (Causality). The SDTM-IG states that there currently is no controlled terminology for this variable. For AEREL there is once again an ODM "hot candidate" (Relationship to study drug), so this is an easy mapping.
Getting up the "CDISC Notes" (using CTRL-H) provides:


SDTM CDISC Note for Variable AE.AEREL

CDISC Notes:
Records the investigator's opinion as to the causality of the event to the treatment. ICH E2A and E2B examples include "NOT RELATED", "UNLIKELY RELATED", "POSSIBLY RELATED", "RELATED". Controlled Terminology may be defined in the future. Check with regulatory authority for population of this variable.

Core: Exp

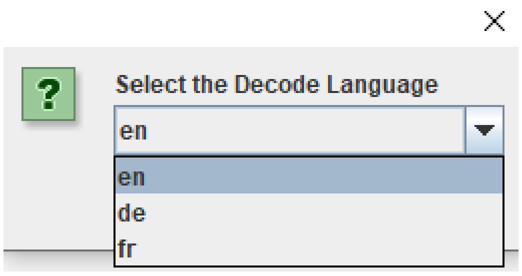
Just for the exercise, we will this time use the ODM values, but as these are coded (coded values 0-3) we want to use the ODM decoded values ("None", "Unlikely", "Possible", "Probable"). So in the dialog, we choose for "Use Study CodeList (from ODM) – Decoded Values":


A CodeList is associated with ODM Item 'Relationship to study drug'

The ODM ItemDef is ruled by a CodeList.
You can either keep using the associated ODM CodeList,
or select another CodeList from the define.xml structure.
In the case a CodeList from the SDTM / define.xml is selected,
a CodeList mapping will be necessary.
In all applicable cases, a (new) CodeList reference will be added to the
corresponding ItemDef in the define.xml structure.

○ Use Study CodeList (from ODM) - Coded Values
● Use Study CodeList (from ODM) - Decoded Values
○ Use another CodeList (from define.xml)

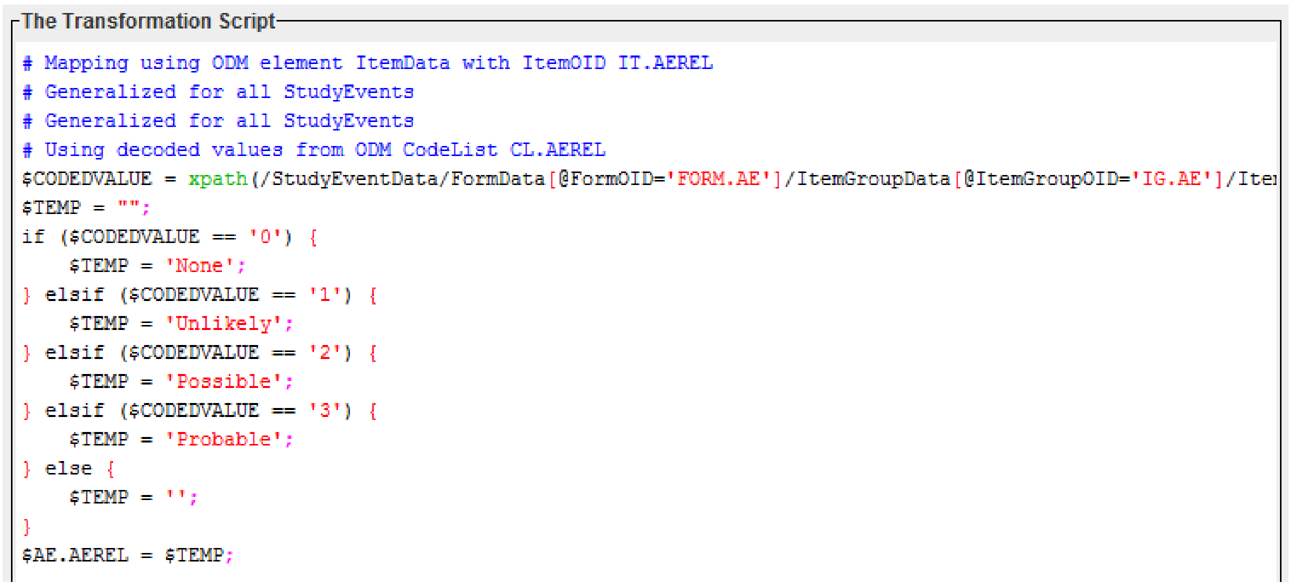
As our ODM has decoded values for different languages, the system now asks us for which language we want to retrieve the decoded values:


Select the Decode Language

en

We choose "English" ("en"). Again, the mapping script is automatically created:

leading to the following mapping script:

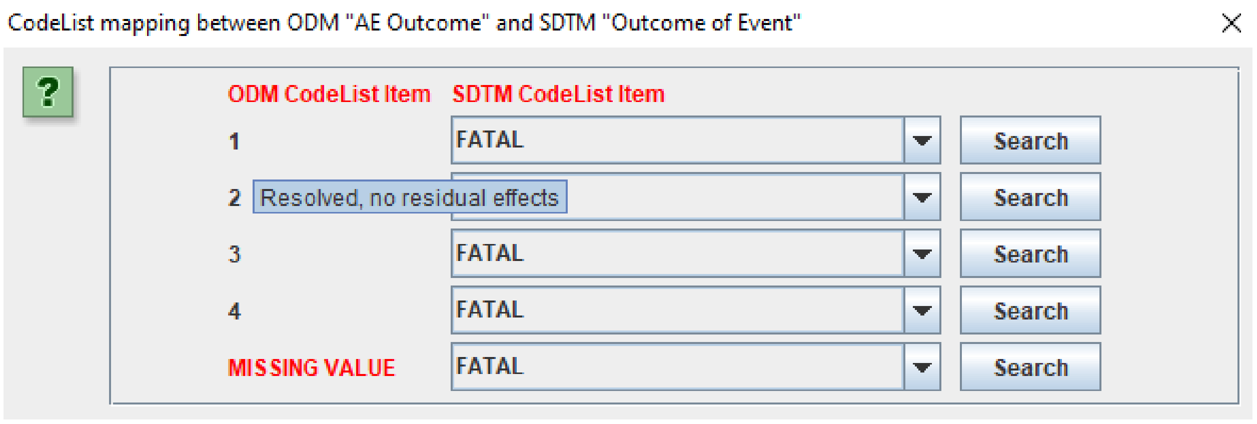

The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.AEREL
# Generalized for all StudyEvents
# Generalized for all StudyEvents
# Using decoded values from ODM CodeList CL.AEREL
$CODEDVALUE = xpath(/StudyEventData/FormData[@FormOID='FORM.AE']/ItemGroupData[@ItemGroupOID='IG.AE']/Ite
$TEMP = "";
if ($CODEDVALUE == '0') {
    $TEMP = 'None';
} elsif ($CODEDVALUE == '1') {
    $TEMP = 'Unlikely';
} elsif ($CODEDVALUE == '2') {
    $TEMP = 'Possible';
} elsif ($CODEDVALUE == '3') {
    $TEMP = 'Probable';
} else {
    $TEMP = '';
}
$AE.AEREL = $TEMP;
```

Now do the same for the SDTM variable AE.AEACNOTH ("Other action taken"). It can be mapped to ItemDef "Actions Taken – Other". Once again, use decoded values from the ODM codelist.

For the SDTM variable AE.AEOUT, there is again an ODM "hot candidate" (ItemDef: Outcome).

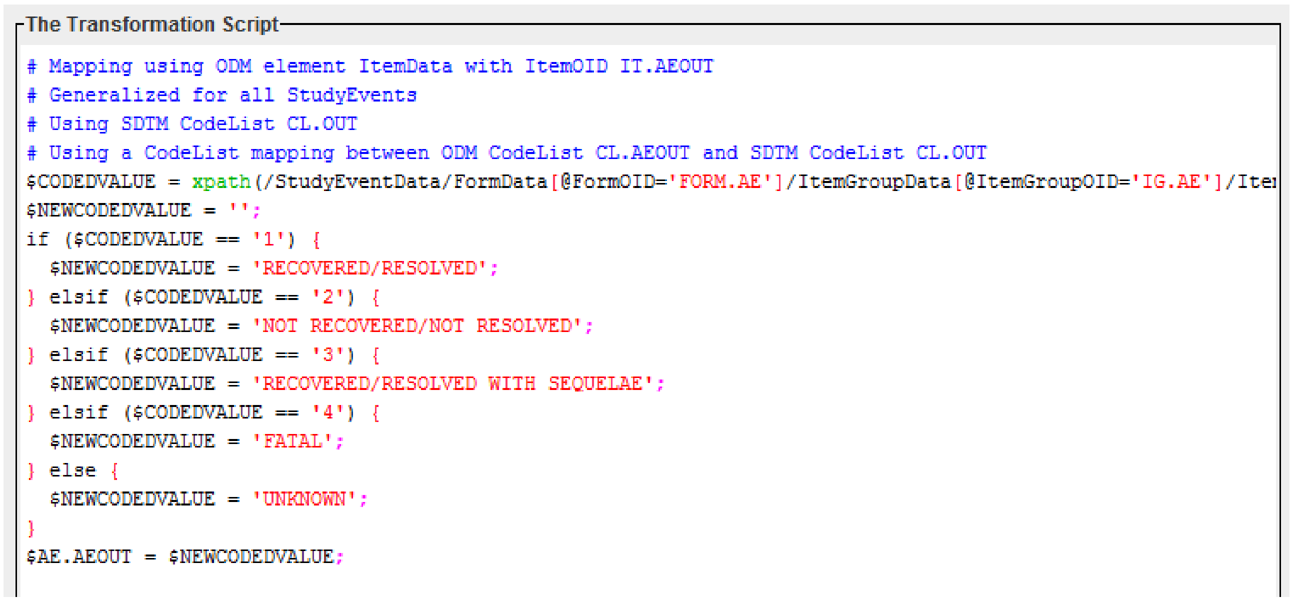
There is an ODM codelist associated, and at the SDTM side, there is already an associated codelist CL.OUT. The usual process leads us to the codelist mapping wizard:


CodeList mapping between ODM "AE Outcome" and SDTM "Outcome of Event"

| ODM CodeList Item | SDTM CodeList Item |
|---|---|
| 1 | FATAL |
| 2 | FATAL |
| 3 | FATAL |
| 4 | FATAL |
| MISSING VALUE | FATAL |

Resolved, no residual effects

and the automatically created mapping script:


The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.AEOUT
# Generalized for all StudyEvents
# Using SDTM CodeList CL.OUT
# Using a CodeList mapping between ODM CodeList CL.AEOUT and SDTM CodeList CL.OUT
$CODEDVALUE = xpath(/StudyEventData/FormData[@FormOID='FORM.AE']/ItemGroupData[@ItemGroupOID='IG.AE']/Ite
$NEWCODEDVALUE = '';
if ($CODEDVALUE == '1') {
  $NEWCODEDVALUE = 'RECOVERED/RESOLVED';
} elsif ($CODEDVALUE == '2') {
  $NEWCODEDVALUE = 'NOT RECOVERED/NOT RESOLVED';
} elsif ($CODEDVALUE == '3') {
  $NEWCODEDVALUE = 'RECOVERED/RESOLVED WITH SEQUELAE';
} elsif ($CODEDVALUE == '4') {
  $NEWCODEDVALUE = 'FATAL';
} else {
  $NEWCODEDVALUE = 'UNKNOWN';
}
$AE.AEOUT = $NEWCODEDVALUE;
```

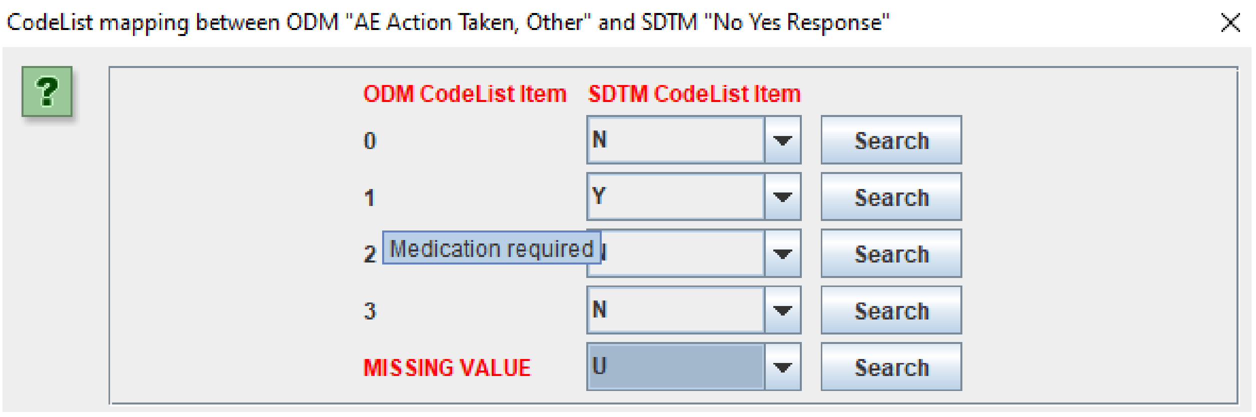
You can now also easily make similar mappings for the following SDTM variables: AE.AESDTH (Results in Death), AE.AEHOSP (Hospitalization), AE.AESLIFE (Life Threatening), and AE.AECONTRT (Concomitant or additional treatment given). In each case, the associated codelist is CL.NY.

Some hints for the ODM items that can be used and the ODM and SDTM codelists involved, are given in the following table:

| ***SDTM Variable*** | ***ODM Item*** | ***ODM Codelist*** | ***Define.xml codelist*** |
|---|---|---|---|
| AESDTH (Results in Death) | Outcome (IT.AEOUT) | CL.AEOUT | CL.NY |
| AEHOSP (Requires or Prolongs Hospitalization) | Actions taken, other (IT.AECONTRT) | CL.AECONTRT | CL.NY |
| AESLIFE | Severity (IT.AESEV) | CL.AESEV | CL.NY |

| ***SDTM Variable*** | ***ODM Item*** | ***ODM Codelist*** | ***Define.xml codelist*** |
|---|---|---|---|
| (Is Life Threatening) | | | |
| AECONTRT (Concomitant or Additional Trtmnt Given) | Actions taken, other (IT.AECONTRT) | CL.AECONTRT | CL.NY |

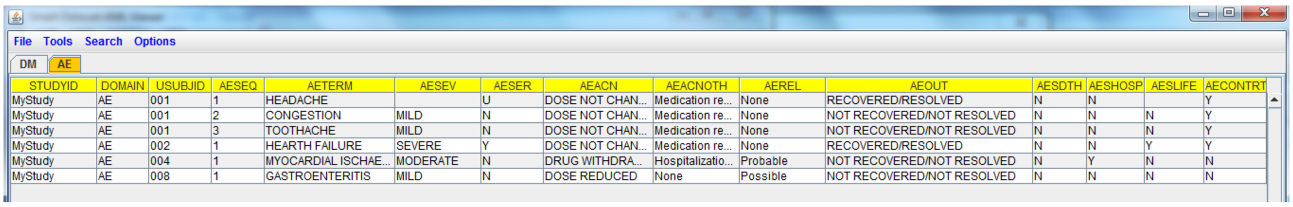
For example, for AECONTRT (Concomitant or Additional Trtmnt Given) the mapping to the ODM "Actions taken, other", can look like:


CodeList mapping between ODM "AE Action Taken, Other" and SDTM "No Yes Response"

| ODM CodeList Item | SDTM CodeList Item | |
|---|---|---|
| 0 | N | Search |
| 1 | Y | Search |
| 2 Medication required | [illegible] | Search |
| 3 | N | Search |
| MISSING VALUE | U | Search |

Note that it depends on the structure and content of your CRF whether <u>absence</u> of additional treatment should be mapped to either „N" or null for AECONTRT.

Executing the already available mappings using the CDISC Dataset-XML format then leads to:



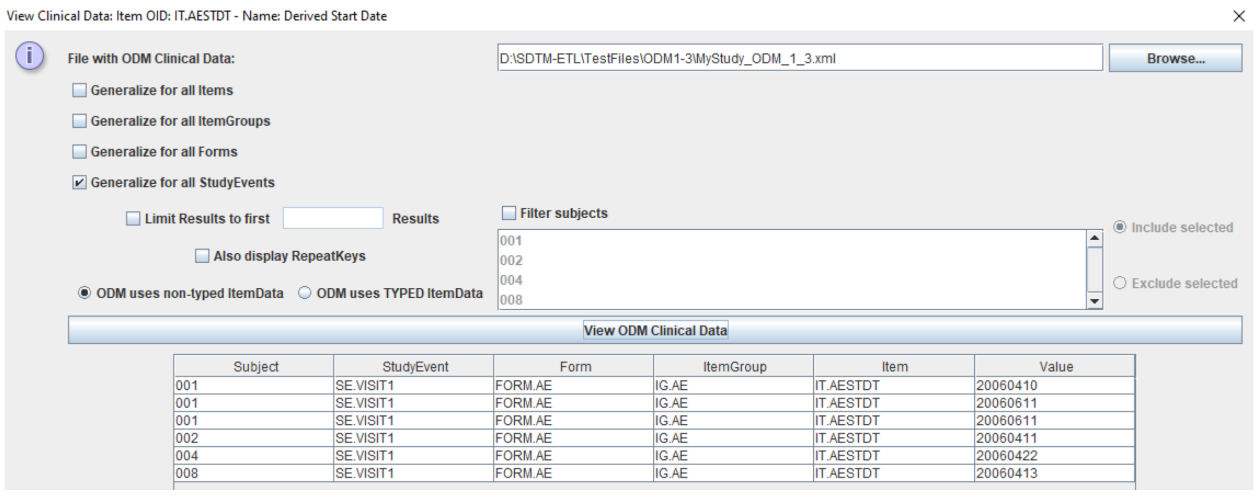
| STUDYID | DOMAIN | USUBJID | AESEQ | AETERM | AESEV | AESER | AEACN | AEACNOTH | AEREL | AEOUT | AESDTH | AESHOSP | AESLIFE | AECONTRT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MyStudy | AE | 001 | 1 | HEADACHE | | U | DOSE NOT CHAN... | Medication re... | None | RECOVERED/RESOLVED | N | N | | Y |
| MyStudy | AE | 001 | 2 | CONGESTION | MILD | N | DOSE NOT CHAN... | Medication re... | None | NOT RECOVERED/NOT RESOLVED | N | N | N | Y |
| MyStudy | AE | 001 | 3 | TOOTHACHE | MILD | N | DOSE NOT CHAN... | Medication re... | None | NOT RECOVERED/NOT RESOLVED | N | N | N | Y |
| MyStudy | AE | 002 | 1 | HEARTH FAILURE | SEVERE | Y | DOSE NOT CHAN... | Medication re... | None | RECOVERED/RESOLVED | N | N | Y | Y |
| MyStudy | AE | 004 | 1 | MYOCARDIAL ISCHAE... | MODERATE | N | DRUG WITHDRA... | Hospitalizatio... | Probable | NOT RECOVERED/NOT RESOLVED | N | Y | N | N |
| MyStudy | AE | 008 | 1 | GASTROENTERITIS | MILD | N | DOSE REDUCED | None | Possible | NOT RECOVERED/NOT RESOLVED | N | N | N | N |

## <u>Timing variables</u>

The SDTM variables AESTDTC (AE start date), AEENDTC (AE end date), AESTDY (Study day of start AE), AEENDY (Study day of end AE), and AEDUR (AE duration) require somewhat more attention. For the first two, the information is available in the ODM ("Derived Start Date"[11] and "Derived Stop Date"). The next two need to be calculated using e.g. the date of the first visit as study day 1. The last (AE duration) should only be submitted (according to the SDTM-IG) in case it was <u>collected</u> as a duration, so it should not be derived. However, just for the exercise, we will calculate it anyway.

In SDTM, dates should be given in ISO-8601 format (YYYY-MM-DD). If we look however at the ODM clinical data for e.g. "Derived Start Date", we see that these are not in the required ISO-8601 format:

[11]It is named „derived" as it comes from the combination of „start year" (Item OID IT.AESTYR), „start month" (IT.AESTMON) and „start day" (IT.AESTDAY)


View Clinical Data: Item OID: IT.AESTDT - Name: Derived Start Date

File with ODM Clinical Data: D:\SDTM-ETL\TestFiles\ODM1-3\MyStudy_ODM_1_3.xml

- [ ] Generalize for all Items
- [ ] Generalize for all ItemGroups
- [ ] Generalize for all Forms
- [✓] Generalize for all StudyEvents

[ ] Limit Results to first Results

[ ] Also display RepeatKeys

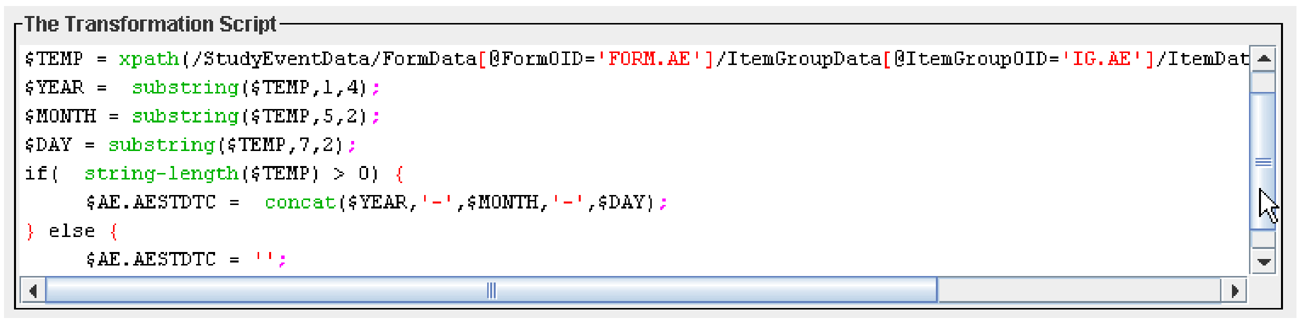
[● ODM uses non-typed ItemData] ○ ODM uses TYPED ItemData

[ ] Filter subjects: 001, 002, 004, 008 — [● Include selected] ○ Exclude selected

| Subject | StudyEvent | Form | ItemGroup | Item | Value |
|---|---|---|---|---|---|
| 001 | SE.VISIT1 | FORM.AE | IG.AE | IT.AESTDT | 20060410 |
| 001 | SE.VISIT1 | FORM.AE | IG.AE | IT.AESTDT | 20060611 |
| 001 | SE.VISIT1 | FORM.AE | IG.AE | IT.AESTDT | 20060611 |
| 002 | SE.VISIT1 | FORM.AE | IG.AE | IT.AESTDT | 20060411 |
| 004 | SE.VISIT1 | FORM.AE | IG.AE | IT.AESTDT | 20060422 |
| 008 | SE.VISIT1 | FORM.AE | IG.AE | IT.AESTDT | 20060413 |

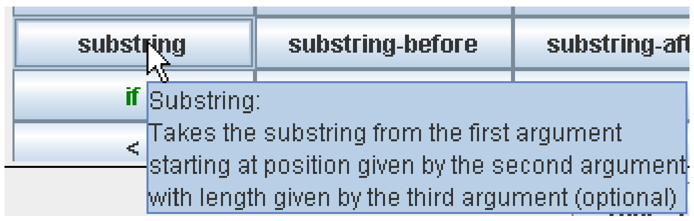
The ISO-format is YYYY-MM-DD, e.g. for the first date it should be: 2006-04-10.

However, due to the numerous script functions that we have available, it is pretty easy to make a transformation, after having performed drag-and-drop from the ODM item "Derived Start Date" to the SDTM cell AESTDTC. For example:


The Transformation Script

```
$TEMP = xpath(/StudyEventData/FormData[@FormOID='FORM.AE']/ItemGroupData[@ItemGroupOID='IG.AE']/ItemDat
$YEAR =  substring($TEMP,1,4);
$MONTH = substring($TEMP,5,2);
$DAY = substring($TEMP,7,2);
if(  string-length($TEMP) > 0) {
     $AE.AESTDTC =  concat($YEAR,'-',$MONTH,'-',$DAY);
} else {
     $AE.AESTDTC = '';
```

The first line retrieves the value from the ODM clinical data.
The second line takes the first four characters from the retrieved value, the third line the next two characters, and the fourth line the last two characters.
If you do not understand how the "substring" function works, just look for the button near the bottom of the screen, and hold the mouse over it. A tooltip shows up:



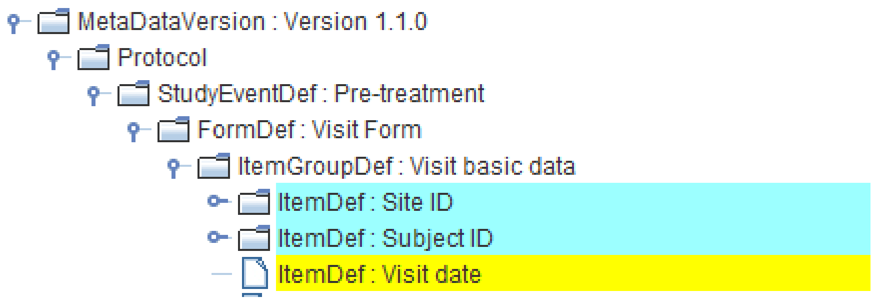
Substring:
Takes the substring from the first argument starting at position given by the second argument with length given by the third argument (optional)

The fifth line in the script then sets up an if-else structure. In case there is a value for the date, it is reassembled from the individual parts using the dash as separator between year, month and day. The "else" part is necessary as otherwise we would get "--" in case there is a missing value for the date.

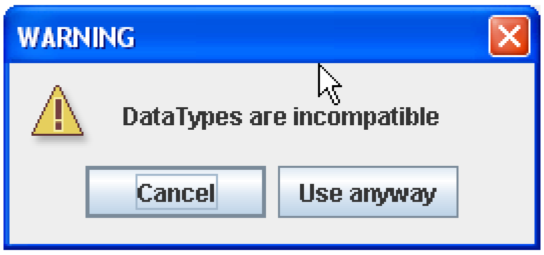
For the "Study day of AE start" (AE.AESTDY), we need to do some date handling.We will take the visit date of the first visit as study day 1 (see remark [12]), and then calculate the value for

[12]In SDTM study days (relative to the reference start date) can either be positive values, or negative values, but NEVER 0 (FDA math ...). This always needs to be taken into account when calculating study days.

AE.AESTDY from this and the AE start date. For the former, we can use drag-and-drop from the ODM, this although the datatypes of source and target do not match (red traffic light).

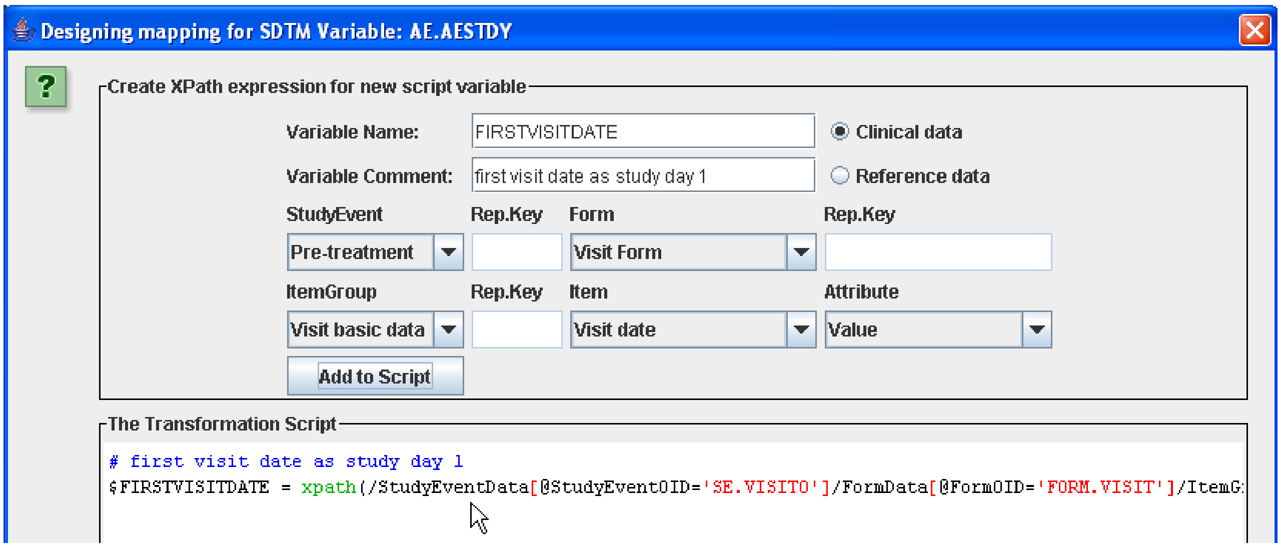

MetaDataVersion : Version 1.1.0
Protocol
StudyEventDef : Pre-treatment
FormDef : Visit Form
ItemGroupDef : Visit basic data
ItemDef : Site ID
ItemDef : Subject ID
ItemDef : Visit date

We then get the following warning message:


WARNING

DataTypes are incompatible

In this case, we should have the checkbox "Generalize for all StudyEvents" unchecked!
We can however also easily use the mapping selectors in the mapping editor:


Designing mapping for SDTM Variable: AE.AESTDY

Create XPath expression for new script variable

Variable Name: FIRSTVISITDATE [✓ Clinical data] Reference data

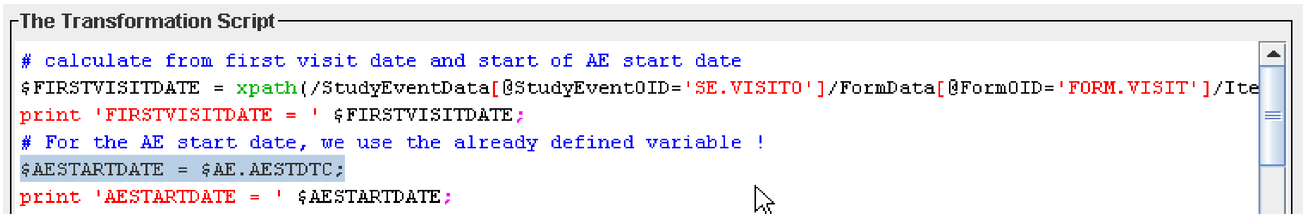
Variable Comment: first visit date as study day 1

| StudyEvent | Rep.Key | Form | Rep.Key |
|---|---|---|---|
| Pre-treatment | | Visit Form | |
| **ItemGroup** | **Rep.Key** | **Item** | **Attribute** |
| Visit basic data | | Visit date | Value |

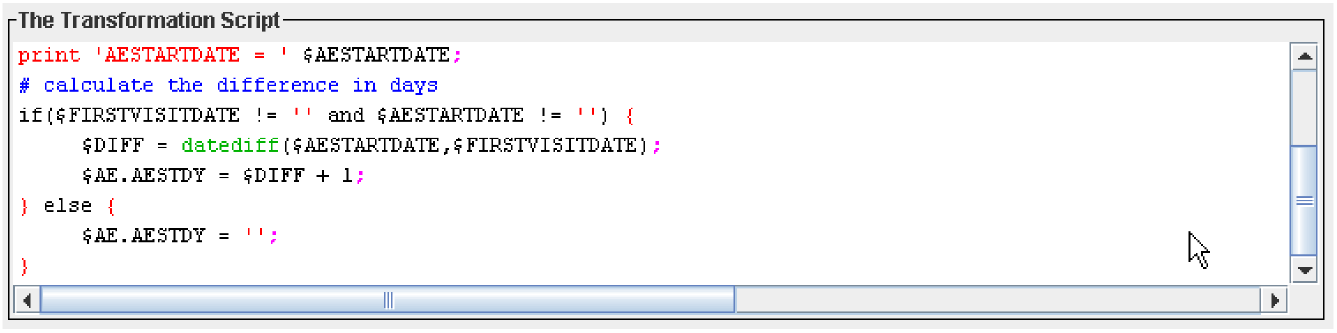
The Transformation Script

```
# first visit date as study day 1
$FIRSTVISITDATE = xpath(/StudyEventData[@StudyEventOID='SE.VISIT0']/FormData[@FormOID='FORM.VISIT']/ItemG
```

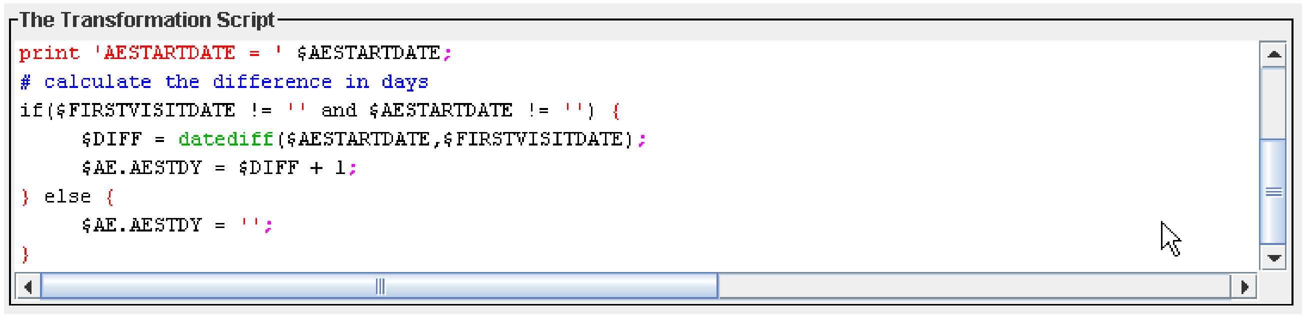
So we defined the first visit date as our reference day.
Now we define the AE start date, **using the already defined variable $AE.AESTDTC**. This may look strange at first, but we may always use previously defined (i.e. in a cell before the current one in the same row) scripting variable in "read" mode. So we write:


The Transformation Script
```
# calculate from first visit date and start of AE start date
$FIRSTVISITDATE = xpath(/StudyEventData[@StudyEventOID='SE.VISIT0']/FormData[@FormOID='FORM.VISIT']/Ite
print 'FIRSTVISITDATE = ' $FIRSTVISITDATE;
# For the AE start date, we use the already defined variable !
$AESTARTDATE = $AE.AESTDTC;
print 'AESTARTDATE = ' $AESTARTDATE;
```

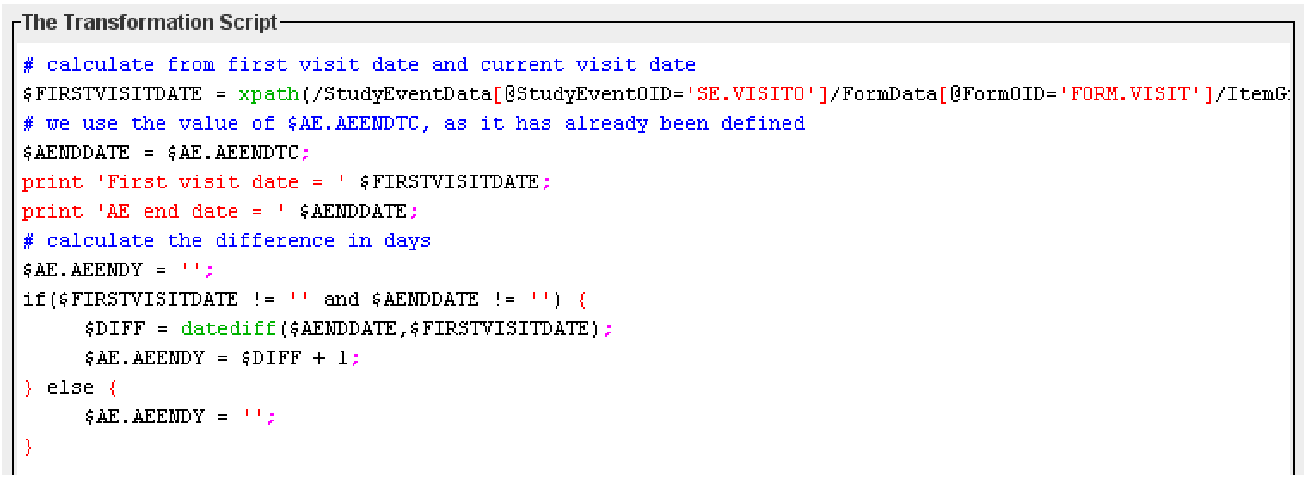
Also remark that we added some "print" statements for testing.
We know that both dates are in ISO-8601 format, so we can now use the "datediff" function to calculate the number of days between both dates, and from that, retrieve the study day of AE start:


The Transformation Script
```
print 'AESTARTDATE = ' $AESTARTDATE;
# calculate the difference in days
if($FIRSTVISITDATE != '' and $AESTARTDATE != '') {
     $DIFF = datediff($AESTARTDATE,$FIRSTVISITDATE);
     $AE.AESTDY = $DIFF + 1;
} else {
     $AE.AESTDY = '';
}
```

We need to take into account missing values for the AE start date. If we don't, the "datediff" function will fail and raise an exception (i.e. the execution of the script will crash).
Also we need to add 1 to the result, as the first study day is day 1, and not day 0.


The Transformation Script
```
print 'AESTARTDATE = ' $AESTARTDATE;
# calculate the difference in days
if($FIRSTVISITDATE != '' and $AESTARTDATE != '') {
     $DIFF = datediff($AESTARTDATE,$FIRSTVISITDATE);
     $AE.AESTDY = $DIFF + 1;
} else {
     $AE.AESTDY = '';
}
```

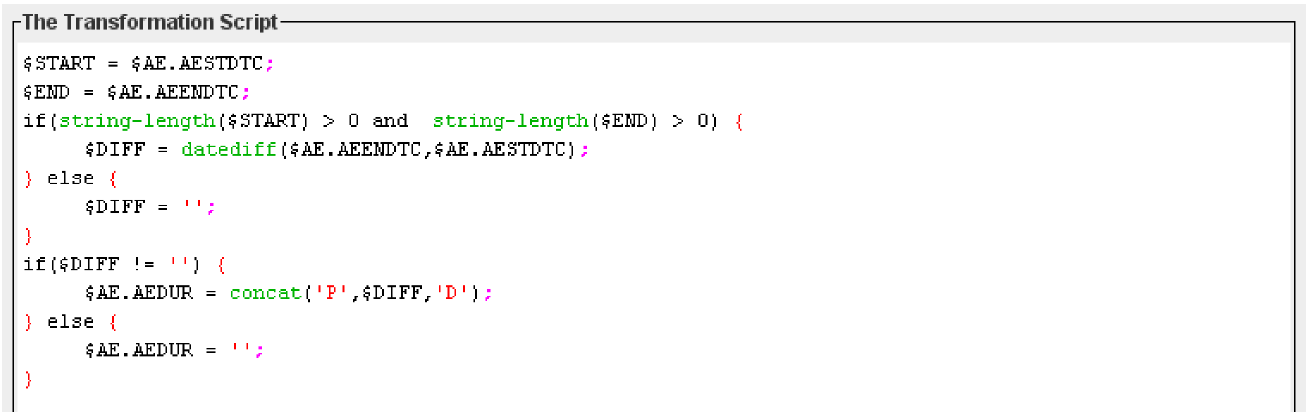
We can do similarly for the "Study day of AE end" (AE.AENDY):


The Transformation Script
```
# calculate from first visit date and current visit date
$FIRSTVISITDATE = xpath(/StudyEventData[@StudyEventOID='SE.VISIT0']/FormData[@FormOID='FORM.VISIT']/ItemG:
# we use the value of $AE.AEENDTC, as it has already been defined
$AENDDATE = $AE.AEENDTC;
print 'First visit date = ' $FIRSTVISITDATE;
print 'AE end date = ' $AENDDATE;
# calculate the difference in days
$AE.AEENDY = '';
if($FIRSTVISITDATE != '' and $AENDDATE != '') {
     $DIFF = datediff($AENDDATE,$FIRSTVISITDATE);
     $AE.AEENDY = $DIFF + 1;
} else {
     $AE.AEENDY = '';
}
```

Here again, we reuse (but "read-only") $AE.AEENDTC, for the calculation of the study day.

There is even a simpler way that we will learn about a bit later. We will need "reference start date" (from DM-RFSTDTC) and "reference end date" (from DM-RFENDTC) very often (in almost each domain), so we will create some "subject global variables" that we can then use over and over again in "read only" mode. For more details, see the tutorial „creating and working with global variables".

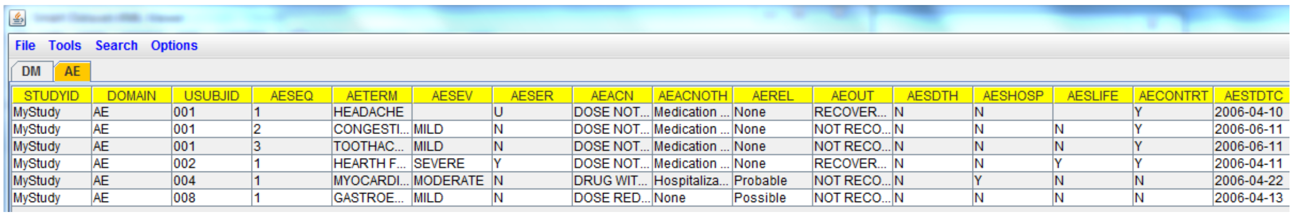
The last SDTM variable we want to provide a mapping for is AE.AEDUR (duration of the AE event). According to the SDTM-IG, it should only be submitted if it was collected as a duration on the CRF. However, for the sake of the exercise, we will derive it here.
As we do already have the start and end date, this is very easy, and we can use the datediff() function again. We only need to take missing values into account.
The script then looks like:


The Transformation Script

```
$START = $AE.AESTDTC;
$END = $AE.AEENDTC;
if(string-length($START) > 0 and  string-length($END) > 0) {
     $DIFF = datediff($AE.AEENDTC,$AE.AESTDTC);
} else {
     $DIFF = '';
}
if($DIFF != '') {
     $AE.AEDUR = concat('P',$DIFF,'D');
} else {
     $AE.AEDUR = '';
}
```

The three last lines take care that the date difference (in days) is put into ISO-8601 "duration" format.

Now, let us once again execute the mapping on a file with clinical data. We again use the menu "Transform -> Generate Transformation Code for CDISC Dataset-XML -> Execute Transformation Code". The result as inspected using the „Smart Dataset-XML Viewer" is:



| STUDYID | DOMAIN | USUBJID | AESEQ | AETERM | AESEV | AESER | AEACN | AEACNOTH | AEREL | AEOUT | AESDTH | AESHOSP | AESLIFE | AECONTRT | AESTDTC |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MyStudy | AE | 001 | 1 | HEADACHE |  | U | DOSE NOT... | Medication ... | None | RECOVER... | N | N |  | Y | 2006-04-10 |
| MyStudy | AE | 001 | 2 | CONGESTI... | MILD | N | DOSE NOT... | Medication ... | None | NOT RECO... | N | N | N | Y | 2006-06-11 |
| MyStudy | AE | 001 | 3 | TOOTHAC... | MILD | N | DOSE NOT... | Medication ... | None | NOT RECO... | N | N | N | Y | 2006-06-11 |
| MyStudy | AE | 002 | 1 | HEARTH F... | SEVERE | Y | DOSE NOT... | Medication ... | None | RECOVER... | N | N | Y | Y | 2006-04-11 |
| MyStudy | AE | 004 | 1 | MYOCARDI... | MODERATE | N | DRUG WIT... | Hospitaliza... | Probable | NOT RECO... | N | Y | N | N | 2006-04-22 |
| MyStudy | AE | 008 | 1 | GASTROE... | MILD | N | DOSE RED... | None | Possible | NOT RECO... | N | N | N | N | 2006-04-13 |

Remark again that AEDUR should normally not be calculated during the mapping, but only be provided when it was collected on the CRF.
We did calculate it here just for the sake of the exercise.