

CDISC Implementation Step by Step: A Real World Example

Sy Truong, Meta-Xceed, Inc., Milpitas, CA
Patricia Gerend, Genentech, Inc., South San Francisco, CA

ABSTRACT

Implementation of the CDISC SDTM data model is no longer a theoretical academic exercise but is now entering the real world since the FDA issued a guidance in April, 2006, specifying its use. This paper will walk you through the lessons learned from implementations of CDISC SDTM version 3.1.1. It will cover both technical challenges along with methodologies and processes. Some of the topics covered include:

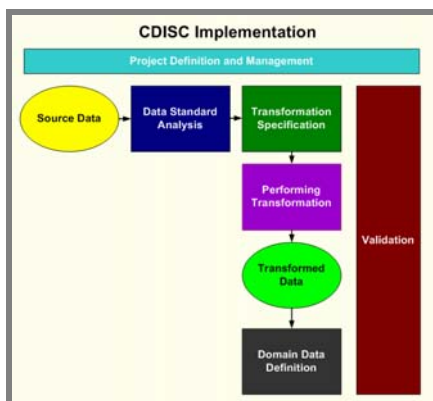
- Project Definition, Plan and Management
- Data Standards Analysis and Review
- Data Transformation Specification and Definition
- Transforming Data to Standards
- Review and Validation of Transformations and Standards
- Domain Documentation for DEFINE.PDF and DEFINE.XML

The regulatory requirements are likely to include CDISC in the near future and the benefits of industry-wide standardization are obvious. It is therefore wise and prudent to implement these standards with techniques and processes refined from lessons learned from real life implementations.

INTRODUCTION

CDISC (Clinical Data Interchange Standards Consortium) standards have been in development for many years. There have been major structural changes to the recommended standards from version 2 to 3. It is still an evolving process but it has reached a point of critical mass such that organizations are recognizing the benefits of taking the proposed standard data model out of the theoretical realm and putting it into real life applications. The complexity of clinical data coupled with technologies involved can make implementation of a new standard challenging. This paper will explore the pitfalls of the CDISC SDTM (Submission Data Tabulation Model) transformation and present methodologies and technologies that make the transformation of nonstandard data into CDISC efficient and accurate.

There are some tasks within the process that can be applied asynchronously, but the majority of the steps depend on each other and therefore follow a sequence. The process is described below:



It is important to have a clear vision of the processes for the project before you start and to be aware that the effort is resource-intensive. This provides the ability to resource and plan for all the processes and enables adherence to deadlines and budgets. The organization and planning for this undertaking is an essential first step towards an effective implementation.

PROJECT MANAGEMENT

Before any data is transformed or any programs are developed, a project manager needs to clearly define the project for CDISC implementation. This is an essential step which will clarify the vision for the entire team and will galvanize the organization into committing to this endeavor. The project definition and plan works on multiple levels by providing a practical understanding of the steps required to create a consensus building team effort. This can avoid the potential political battles which do arise among distinct departments within an organization. The following steps will walk you through the project planning stage.

STEP 1: DEFINE SCOPE – The project scope should be clearly stated in a document. This does not have to be long and can be as short as one paragraph. The purpose of this is to clearly define the boundaries of the project since without this definition the project has tendencies towards scope creep. It can therefore potentially exceed your resource budget. Some of the parameters to be considered for the scope of the project include:

- **Pilot** – For an initial project, it is a good idea to pilot this on one or two studies before implementing this broadly. The specific study should be selected based on being representative of other studies likely to be converted to CDISC.
- **Scope** – This could be scoped as a limited roll out of a new standard or a global implementation for the entire organization. This also requires quantifying details such as how many studies are involved and which group(s) will be affected. Not only does this identify resources in the area of programming and validation, but it also determines the training required.
- **Standard Audience** – The scope should clearly identify the user groups who will be affected by this standard. It can be limited to the SAS programming and biostatistics group, or it can have implications for data management, clinical operations, publishing, regulatory affairs, and electronic submission groups.
- **Validation** – The formality of the validation is dictated by the risk analysis. The scope defines if the project includes validation, or if this would be part of a separate task.
- **Documentation** – Data definition documentation is commonly generated as part of an electronic submission. It is a task that is performed with a CDISC implementation. The scope would identify if the data definition is part of the project or considered another project altogether.
- **Establishing Standards** – The project may be used to establish a future set of standards. The scope should identify if establishment of global standards is expected or if it is just a project specific implementation.

The scope document is a form of a requirements document which will help identify the goals for the project. It can also be used as a communication method to managers and team members to set the appropriate level of expectations.

STEP 2: IDENTIFY TASKS – Capture all the tasks that are required to transform your data to CDISC. This may vary depending on the scope and goals of the project. If the project is a pilot, for example, the tasks would be limited compared to a global implementation. The following is an example list of a subset of tasks along with the associated estimated time of performance.

Data Transformation to CDISC

Project Tasks	Estimated Work Units
Initial review of study's data standards including checking all data attributes for consistency. Generation of necessary reports for documentation and communication.	17
Reconciliation of internal data standards deviations with organization's managers.	17

Project Tasks	Estimated Work Units
Data integrity review including invalid dates, format codes and other potential data errors. Generation of reports documenting any potential data discrepancies.	17
Initial data review against a prescribed set of CDISC SDTM requirements and guidelines. Generation of a report with recommendations on the initial application of CDISC SDTM standards.	17
Decisions on implementing initial CDISC SDTM data review and identification of tasks to be performed.	17
Performance of a thorough review of all data and associated attributes. Identification of all recommended transformation requirements. This is documented in a transformation requirement specification.	42
Creation of transformation models based on the transformation specifications for each data set.	25
Generation of the code to perform transformations for each transformation model.	50
Generation of test verification scripts to verify and document each transformation program against the transformation requirements specification.	42
Performance of testing and validation of all transformations for data integrity. Reconciliation and resolution of associated deviations.	42
Execution of the transformation programs to convert the data into CDISC SDTM format.	25
Performance of data standard review and data integrity review of newly created transposed data in CDISC SDTM format.	17
Documentation in summary reports of all transformations. This also includes a summary of all test cases explaining any deviation and how it was resolved.	17
Project management activities including coordinating meetings and summarizing status updates for more effective client communication pertaining to CDISC SDTM data.	25
Total Estimates	370

This initial step is only meant as an estimate and will require periodic updates as the project progresses. It should be detailed enough so that team members who are involved with the project have a picture of and appreciation for the project. The experience of the project manager will determine the accuracy of the tasks and associated time estimates. In this example, it has not been specified how many person hours this will be but in the real world, this will more closely reflect your team's efforts in estimated hours.

This document is used to communicate with all team members that will potentially work on the project. Feedback should be incorporated to make the identified tasks and the estimates as accurate as possible.

STEP 3: PROJECT PLAN – Once the tasks have been clearly documented, the list of tasks will be expanded into a project plan. The project plan is an extension of the task list including more of the following types of information:

- *Project Tasks* – Tasks are grouped by function. This is usually determined by the skills required to perform the task. This can correlate to individuals involved or whole departments. Groups of tasks can also be determined by the chronological order in which they are to be performed. If a series of steps requires that they be done one after another, they should be grouped.

- *Tasks Assignments* – Once the tasks have been grouped by function, they are assigned to a department, manager or an individual. The logistics of this depends on the SOPs of your organization. This however needs to be clearly defined for planning and budgeting purposes.
- *Schedules of Tasks* – A time line is drafted noting at a high level when important deliverables or milestones are met. The titles of the tasks are the same as the title for the group of tasks. This will allow users to link back to the list of tasks to understand the details from the calendar. The schedule is also shown in calendar format for ease of planning.

A subset and sample of the project plan is shown here:

Study ABC1234 CDISC Transformation Project Plan											
<p>Overview This project plan will detail some of the tasks involved in transforming the source data of study ABC1234 into CDISC SDTM in preparation for electronic submission. The proposed time lines are intended as samples which can be adjusted to reflect project priorities.</p>											
<p>Project Tasks</p> <p>The following tasks are organized into groups which have some dependency. They are therefore organized in chronological order.</p>											
<p>1. Data Review</p> <ol style="list-style-type: none"> 1. Evaluate variable attributes differences within internal data of ABC1234 2. Evaluate variable attributes between ABC1234 compared to ACME Standards 3. Evaluate ABC1234 differences and similarities with CDISC SDTM v3.1.1 4. Evaluate potential matches of ABC1234 variable names and labels against CDISC SDTM v3.1.1 5. Evaluate ABC1234 against CDISC formats 6. Generate metadata documentation of the original source data from ABC1234 											
<p>2. Data Transformation Specifications</p> <ol style="list-style-type: none"> 1. Perform a thorough review of all data and associated attributes against CDISC SDTM v3.1.1. Identify all recommended transformation requirements. Document this in a transformation requirements specification. 2. Create transformation models based on the transformation specifications for each data domain. 3. Distribute transformation models for review and feedback. 4. Update the specification to reflect feedback from review 											
<p>Task Assignments</p> <table border="1"> <thead> <tr> <th>Project Tasks</th> <th>Project Manager</th> <th>Team Managers</th> </tr> </thead> <tbody> <tr> <td>Data Review</td> <td>James Brown, Director of Data Management</td> <td>Alicia Keys Billy Joel Joe Jackson</td> </tr> <tr> <td>Data Transformation Specification</td> <td>Janet Jackson, Manager of Biometry</td> <td>Elton John Mariah Carey Christina Aguilera</td> </tr> </tbody> </table>			Project Tasks	Project Manager	Team Managers	Data Review	James Brown, Director of Data Management	Alicia Keys Billy Joel Joe Jackson	Data Transformation Specification	Janet Jackson, Manager of Biometry	Elton John Mariah Carey Christina Aguilera
Project Tasks	Project Manager	Team Managers									
Data Review	James Brown, Director of Data Management	Alicia Keys Billy Joel Joe Jackson									
Data Transformation Specification	Janet Jackson, Manager of Biometry	Elton John Mariah Carey Christina Aguilera									

STEP 4: VALIDATION – Validation is an essential step towards maintaining accuracy and integrity throughout the process. It can be determined to be within or outside the scope of some project. The following list itemizes some of the validation tasks to be performed.

- *Risk Assessment* –Each task or group of tasks is evaluated for risk and the appropriate level of validation is determined.
- *Test Plan* – This documents the testing approach and methodologies to be used during the validation testing. It describes how the testing will be performed and how deviations will be collected and resolved. It also includes test scripts to be used during testing.
- *Summary Results* – This documents the results of the testing. It quantifies the number of deviations and documents how they are to be fixed.

The following example shows a form used to collect tasks and associated risks.

Risk Assessment Title	<i>Risk Assessment of analysis files for sample study.</i>	
Identify the task where the programs reside which contributes to the risk. ⇒	Task Name and Location	<i>Interim Analysis on my server</i>
Identify the groups of programs to see which categories they appear in. ⇒	<input checked="" type="checkbox"/> Analysis Files (20) <input type="checkbox"/> Listings (5) <input type="checkbox"/> Summary Tables (10) <input type="checkbox"/> Graphs (10) <input type="checkbox"/> Edit Checks (5) <input type="checkbox"/> Other:	<i>This is a subset of the analysis files Just as an example.</i>
From the group of programs identified, classify the types of programs. ⇒	Score: 20 <input checked="" type="checkbox"/> Single Use Program in One Study (5) <input type="checkbox"/> Single Use Program (10) <input type="checkbox"/> Multi Use Program in One Study* (20) <input type="checkbox"/> Multi Use Utility or Macro in Multiple Studies* (30) <input type="checkbox"/> Multi Use Utility or Macro in All Studies* (40) <input type="checkbox"/> Other	<i>This is a single use program and it is going to be used in this study only.</i>
What is the likelihood that the program would produce errors or incorrect results? ⇒ <ul style="list-style-type: none"> • Are the specifications not clearly defined? • Does the program use custom logic versus SAS PROCs or standard macros? Score: 10	Score: 5 <input type="checkbox"/> Error Likelihood Detection (0-20)	<i>Since there are some derivations and hard coded values in this code, I will give it some degree of error likelihood.</i>

The test plan can vary depending on the amount of details and level of formality determined by the risk assessment. The following example shows a subset of a more formal test plan. This can be abbreviated to handle transformation tasks that are deemed to be of lower risk.

This document is used both to instruct team members on how to perform the testing and to define how programs are validated. The following is an example of the validation testing approach and execution procedure.

4. VALIDATION TESTING APPROACH

4.1 Operational Qualification (OQ)

4.1.1 OQ will provide assurance that the system meets minimum requirements, required program files are executed, and resulting reports and data produced are operational according to the requirements.

4.1.2 Testers will follow the instructions provided in the Test Scripts to perform the tests as documented in Appendix 1.

4.1.3 All supporting documentation (printouts, attachments, etc.) must be saved and included.

4.2 Summary Report

4.2.1 After all the tests scripts for this validation plan are executed and all deviations have been resolved, a summary report of the test results will be prepared. This summary report will include a discussion of the observed deviations and their resolutions and the storage location of any data not included within the summary report.

5. GENERAL EXECUTION PROCEDURES

5.1 Prerequisites for testing are described in "Test Scripts Setup" in Appendix 1. Once these steps have been completed, the programs for the Test Scripts can be run.

5.2 The testing will be executed either with a batch program or through an interactive visual inspection of reports. For each test, the results will be compared to the expected results either manually or with the aid of comparison tools. The results of such comparisons will be recorded by the tester on the Test Scripts.

5.3 Deviations that occur during testing will be recorded in the Deviation Report, a template for which is included in Appendix 2.

Test Scripts

The format of the test scripts can also vary depending on the formality of your testing. It is important to have each test case contain a unique identifier such as a test case number. This is what a tester and reviewer use to track the test and its associated deviations.

System Name and Version:	Wonder Drug ABC1234 CDISC	Functional Area/Module:	Standardize ABC1234 Data
Test Script Number:	1 (Requirement 4.1)		
Overall Test Objective:	Verify the variable attributes of the existing source data of Wonder Drug ABC1234		
Specific Test Condition:	Tester has read access to input data.		
Test Program Run Location:	Test Area		
Test Program Name(s):	difftest_avf.sas		
Test Script Prerequisites:	None		

Step	Instruction	Expected Result	Actual Result	Initials/Date
1	Right mouse click on test script program and select batch submit.	Script file is executed.		
2	Evaluate log file for errors.	No errors are found.		
3	Evaluate output files to verify that the attributes results match the report that is performed using %difftest as part the summary report.	Output is verified against output.		
Recovery:	Resubmit the program.	Signature/Date		

Step	Instruction	Expected Result	Actual Result	Initials/Date
Final Expected Result:	Verify the variable attributes of the existing source data of Wonder Drug ABC1234.	Actual Result: <input type="checkbox"/> Pass <input type="checkbox"/> Fail		
Comments:		Reviewed By:		

The format presented in the test plan such as the summary report can follow the same format. The examples of this paper only show a subset of the entire test plan and are intended to give you a conceptual understanding so that you can apply the same concepts to all the other parts of the documentation.

STEP 5: TRANSFORMATION SPECIFICATION – The specification for transforming to CDISC standards is a detailed road map that will be referenced and used by all team members during the transformation implementation and review. There can be different technologies used to perform this task. The following example utilizes tools including MS Excel and Transdata™. Dataset transformation is a process in which a set of source datasets and their variables are changed to meet new standard requirements. The following list describes some of the changes that can occur:

1. **Dataset Name** - SAS dataset names must be updated to match SDTM standards, which require them to be no more than 8 characters in length.
2. **Dataset Label** - The descriptive labels of SAS datasets must be modified to conform to SDTM standards.
3. **Variable Name** - Each variable within a SAS dataset has a unique name. Variable names can be the same across different datasets, but if they share the same name, they are generally expected to possess the same attributes. Variable names are no more than 8 characters in length.
4. **Variable Label** - Each variable has an associated label that describes the variable in more detail. Labels are no more than 40 characters in length.
5. **Variable Type** - A variable's type can be either character or numeric.
6. **Variable Length** - A character variable can vary in length from 1 to 200 characters.
7. **Format** - Variable format will be updated.
8. **Yesno** - If the value of the variable is "yes", it will produce a new row with the newly assigned value of the label.
9. **Vertical** - Multiple variables can be assigned to one variable that will produce a row if it has a value.
10. **Combine** - Combine values of multiple source variables into one destination variable.
11. **Drop** - The variable from the source dataset will be dropped when creating the destination data.
12. **Same** - The same variable with all of the same attributes will be kept in the destination data.
13. **Value Change** - This can have either a recoding of values or a formula change. This will change the actual values of the variable.

There may be other types of transformations, but these are the common transformation types that are usually documented in the specification. The transformation specification is stored in an Excel spreadsheet and is organized by tabs. The first tab named "Tables" contains a list of all the source tables. The subsequent tabs contain the transformation specifications for each source dataset as specified in the initial tables tab.

Tables Tab

The Tables tab contains the list of source datasets along with descriptions of how each one transforms into the standard data model. It also records associated data structures such as Relational Records and Supplemental Qualifiers.

ABC1234 Data Transformation

Source Data	CDISC Data Name	SDTM 3.1.1 Label	Related Records	Supplemental Qualifiers
Ae	AE	Adverse Events	AE	AE, CM, EX, DS
Blcancer	DC	Disease Characteristics		DC
Conduct	DV	Protocol Deviations		DV
Death	DS	Disposition	DS	DS
Demog	DM	Demographics Domain Model		DM, EX, DC
Discon	DS	Disposition		DS
Elig	IE	Inclusion/Exclusion Exceptions		MH
Lab	LB	Laboratory Test Results		LB

This will list all the source datasets from the original study. There is not always a one to one transformation. That is, there may be many source datasets used to create one CDISC dataset or one source dataset populating several CDISC datasets. This will act as an index of all the datasets and how they relate to each other. The transformation is not limited to the relationship between source and destination data domains, but also specifies which variables are destined for the “Relational Records” and “Supplemental Qualifiers” datasets. These related data structures are used within SDTM to include data which do not fit perfectly into existing CDISC domains.

Transformation Model Tab

Each source dataset will have a separate corresponding spreadsheet detailing the transformation. The following is an example of an adverse event transformation model tab.

Adverse Event Data Transformation from Study ABC1243 to CDISC SDTM 3.1.1

Variable	Label	Transformation Type	Update To	Domain
PATIENT	Subject ID (Num)	name label length	usubjid label="Unique Subject Identifier" length=\$15	
STUDY	Clinical Study	name label length	studyid label="Study Identifier" length=\$15	
AECONCAU	Causal Con Med	name label length combine	aerelinst label="Relationship to Non-Study Treatment" length=\$140	CM
AECTC	Adverse Event CTC	name label length	aeterm label="Reported Term for the Adverse Event" length=\$150	AE
AECTCORG	Organ System CTC	name label length	aebodsys label="Body System or Organ Class" length=\$30	AE
AECTCOS	Other Specify CTC	Drop		AE
AEDES	AE Description	name label length combine	aeout label="Outcome of Adverse Event" length=\$200	AE
AEDES2	AE Description 2	name label length combine	aeout label="Outcome of Adverse Event" length=\$200	AE

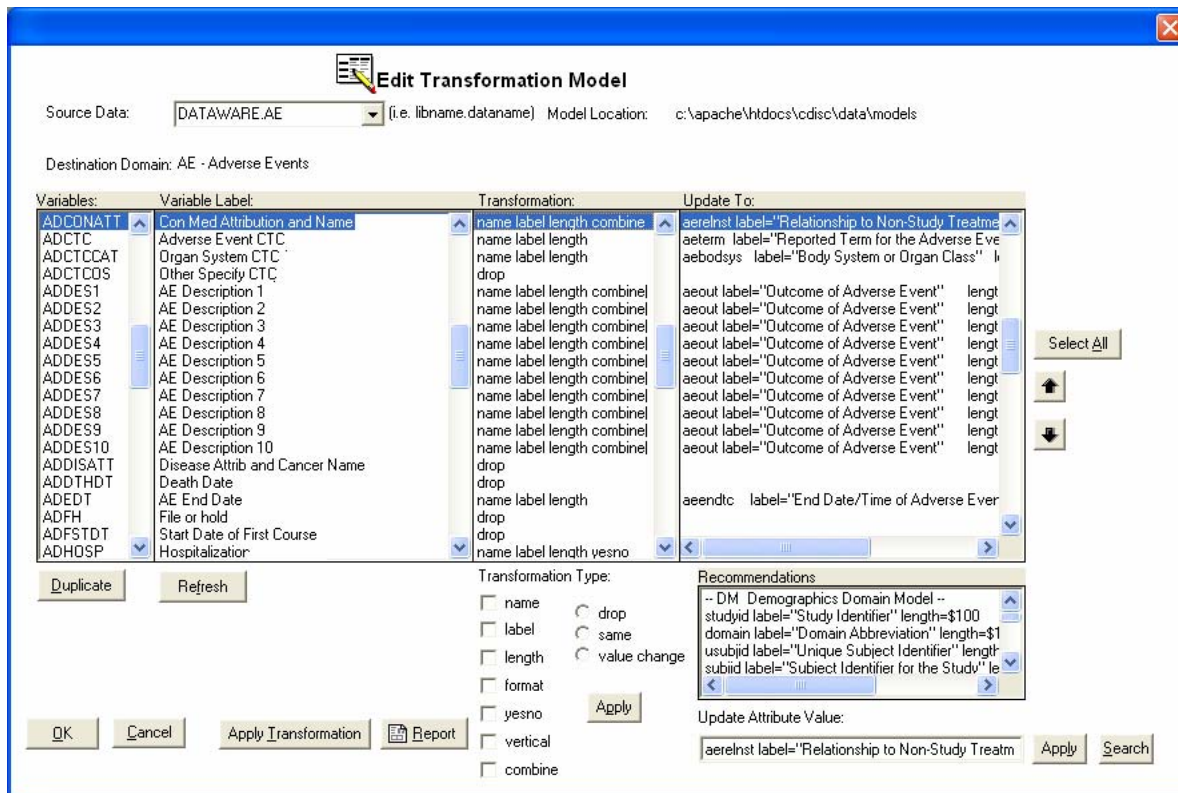
Key

Relational Records
Supplemental Qualifiers
Comments

In this example, the source variable AECTCOS is moved to the supplemental qualifiers structure. Most of the transformations are pretty straightforward attribute changes. However, the transformation of type “combine” will concatenate multiple

source variables into one target variable. Most of these are going towards the AE domain except for the variable AECONCAU which is being transformed into the CM (concomitant medications) domain. This example illustrates how various details of data transformations can be expressed concisely with great detail in the form of a transformation specification.

STEP 6: APPLYING THE TRANSFORMATION – In an ideal world, the specification is completed one time and you apply the transformation according to the specification. In the real world, however, the specification goes through changes throughout the duration of the project. You therefore need to make an executive decision at specified times to apply the transformation even when things are still changing. Because of the dynamic nature of the data, a tool can be very useful since the transformation specification needs to also be dynamic to keep up with the changing data. Changes to transformations have implications for re-coding transformation logic, and manually programming transformations and updates becomes a very resource intensive task. To automate this process, the transformation specification is captured in a SAS dataset and managed with the following screen.



All the source variables and associated labels can be managed and displayed in the left two columns. The type of transformations including the most commonly used ones are listed with check boxes and radio buttons for ease of selection and application. The new attributes of the target variables listed in the specification spreadsheet can also be captured here. Besides being able to edit these attributes, standard attributes from CDISC are also listed as recommendations. The advantages to managing the specifications in this manner compared to Excel include:

1. An audit trail is kept of all changes.
2. The selection choices of transformation type and target attributes make it easier to generate standardized transformations.
3. Transformation logic coding and algorithms can be generated directly from these definitions.
4. A refresh of the source variables can be applied against physical datasets to keep up with changing data.

Program Transformation

Once the transformation specification has been clearly defined and updated against the data, the program(s) must be written to perform these transformations. A sample program may look like this:

```

*****;
* Program: trans_ae.sas
* Path: c:\temp\
* Description: Transform Adverse Events data
*              from DATAWARE.AE to STDMLIB.AE
* By: Sy Truong, 01/21/2006, 3:49:13 pm
*****;
libname DATAWARE "C:\APACHE\HTDOCS\CDISC\DATA";
libname STDMLIB "C:\APACHE\HTDOCS\CDISC\DATA\SDTM";
data STDMLIB.AE (label="Adverse Events"
                );
    set DATAWARE.AE;
    retain obs 1;
    *** Define new variable: aereinst that combined by old variables: aeconcau aerelet;
    attrib aereinst label="Relationship to Non-Study Treatment" length=$140;
    aereinst = trim(trim(aeconcau) || ' ' || aerelet);
    drop aeconcau aerelet;
    *** Define new variable: aeout that combined by old variables: addes addes2;
    attrib aeout label="Outcome of Adverse Event" length=$200;
    aeout = trim(trim(addes) || delimit_aeout0 || addes2) || delimit_aeout1;
    drop delimit_aeout0 delimit_aeout1 addes addes2;
run;

```

There are many different types of transformations. The comment and simple examples described above demonstrate the updates of attributes and assignment to the correct CDISC variables. There are cases where more elaborate transformations are required. The amount of time required to transform each variable in multiple ways makes this step one of the most resource intensive steps.

STEP 7: SPECIAL PURPOSE DOMAIN – CDISC has several special purpose domains. Among these are three named SUPPQUAL, RELREC and CO.

- SUPPQUAL - The Supplemental Qualifiers domain is used to capture non-standard variables and their association to parent records in domains, capturing values for variables not presently included in the general observation-class models.
- RELREC - The Related Records domain is used to describe relationships between records in two (or more) datasets. This includes such records as an “event” record, “intervention” record, or a “finding” record.
- CO - The Comments special-purpose domain is a fixed domain that provides a solution for submitting free-text comments related to data in one or more domains which are collected on a separate CRF page dedicated to comments.

These three are similar in structure and capture values that are related to the main domains.

Supplemental Qualifiers

An example of the SUPPQUAL is shown here:

Preview of Dataset Name: SUPPQUAL

Obs	Study Identifier	Related Domain Abbreviation	Unique Subject Identifier	Identifier Variable	Identifier Variable Value	Qualifier Variable Name	Qualifier Variable Label	Data Value	Origin
1	23423	AE	1803	seqnum	64001	ae	ADVERSE EXPERIENCE		derived
2	23423	AE	101	seqnum	64007	ae	ADVERSE EXPERIENCE	(R) PLEURAL EFFUSION	derived
3	23423	AE	1102	seqnum	64003	ae	ADVERSE EXPERIENCE	(SOB) SHORTNESS OF BREATH	derived
4	23423	AE	1103	seqnum	64001	ae	ADVERSE EXPERIENCE	ABDOMINAL CRAMPING	derived
5	23423	AE	102	seqnum	64009	ae	ADVERSE EXPERIENCE	ABDOMINAL DISTENTION	derived

Most data operationally used in clinical trials are not stored in this manner. The “qualifier variable name” is usually one of the columns in the dataset. However, in this case, it is transposed and the column variable is stored as one of the column values. This allows the structure to handle different variables and to be more flexible. However, this may require you to perform a transposition from your source data.

Since the destination value is a character field, the transformation may require additional conversions. Here is an example code segment that performs this type of transformation.

```
*** Transpose the for variable TRTCYC ***;
data work.suppqual;
  set sourcelib.ae;

  *** Convert numeric idvar ***;
  idvar="aerbdtd";
  idvarvan = aerbdtd;
  idvarval = left(trim(put(idvarvan,DATE9.)));
  qvaln = trtcyc;
  qval = left(trim(put(qvaln,best.)));
  usubjidn = patnum;
  rdomain = "AE";
  qnam = "otherae";
  qlabel = "Other Specify Adverse Event";
run;

*** Append data to the final destination SUPPQUAL ***;
data SDTMLIB.suppqual (label = "Supplemental Qualifiers for Adverse Events");
  set SDTMLIB.suppqual_ae
      work.suppqua2;
  if compress(studyid) = '' and compress(usubjid) = '' then delete;
run;
```

This code segment only shows you part of what is happening. It does however illustrate the need to handle transformations one variable at a time and the need for handling different variable types. If there are many datasets with many variables, this type of transformation can cumulatively add up to a big task. Specialized tools can handle such transformations of structures including SUPPQUAL, RELREC and CO.

The following decisions need to be made when working with the SUPPQUAL dataset:

1. Input Dataset – Select all the input datasets from the source location that need to contribute to SUPPQUAL.
2. Input Variables – Select variables that are not part of the main domain but are considered supplemental. These are deemed important enough to be part of the final submission yet do not fit perfectly to the variables within the specified domain.
3. Source Type – Define the type of source of specified variables. This can have values such as CRF, Assigned, or Derived.
4. Related Domain – Determine the related domain to which this dataset pertains.
5. Study Identifier – Document to which study or protocol name and number this data belongs.
6. Identification Variable – Identify what key fields can be used to uniquely identify the selected fields. This can be a sequence variable, group ID, or unique date variable.
7. Unique Subject ID – Identify the variable which contains the unique subject identification value.

The above decisions can be made with the following interface.

SUPPQUAL
Supplemental Qualifiers and Association to Parent Domain

Source Data: C:\Global\Project1\Study1\Source Data

Data Sets: ADVEMX, ADVERSE, ADVERSE2, ADVERSEX, ADVERSE_, ADVERSMX, AE3, AETSTMX, CO

Variable Names: ae, aend, aepagno, aeseqno, bodyname, bodysubc, coscode, doschng, duration

Original Source Variable: CRF

Related Domain: AE - Adverse Events

Study Identifier: (i.e. Protocol 12345)

Identification Variable: (i.e. --SEQ, --GRPID or --DATE)

Unique Subject ID Variable: patienti

Output: LIP2 (libname), SUPPQUAL_AE (dataname) (i.e. libname.dataname)

Merge into one SUPPQUAL dataset

Base Code: c:\documents and settings\sy truong\suppqual_adverse_ae.sas

Buttons: OK, Cancel, Save Code...

Once all the variables are selected, the user can decide upon the origins. The interface provides default values that can assist the user in making quick decisions. Once the user is proficient at making these types of selections, a macro interface can be used for efficient production batch processing.

Related Records

The related records data domain is similar in structure to the supplemental qualifier domain. These variables are found in events, findings, or intervention records. The domains which are identified in these records include:

Interventions

1. Concomitant Medications
2. Exposure
3. Substance Use

Events

1. Adverse Events
2. Disposition
3. Medical History

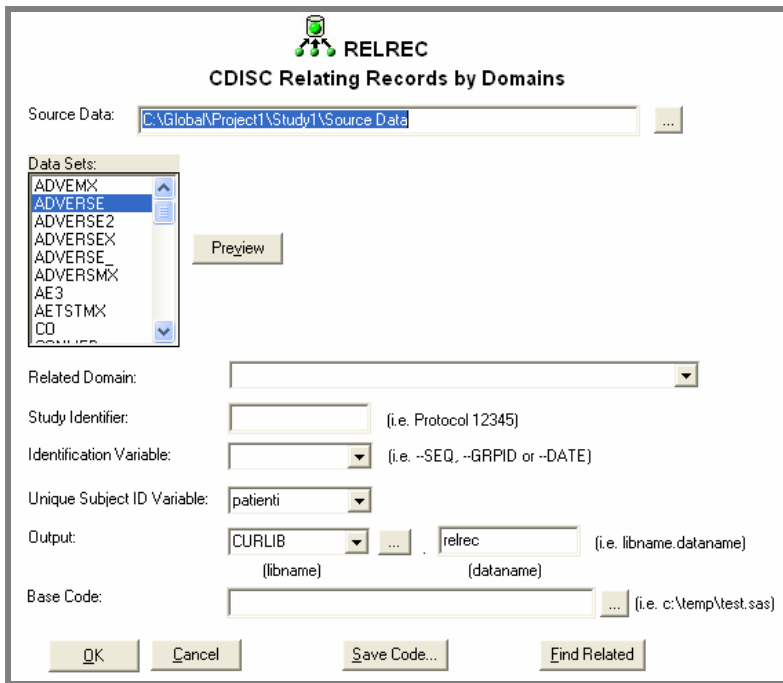
Findings

1. ECG Test Results
2. Inclusion/Exclusion Exceptions
3. Laboratory Test Results
4. Physical Examinations
5. Questionnaires
6. Subject Characteristics
7. Vital Signs

This covers a wide range of fields. The types of fields selected to be related records can be very flexible, but the data structure which is used to store RELREC data is strict. The following decisions need to be made to transpose the data into a related record.

1. Input Dataset – Select all the input datasets from the source location that need to contribute to RELREC.
2. Related Domain – Determine the related domain to which this dataset pertains.
3. Study Identifier – Document to which study or protocol name and number this data belongs.
4. Identification Variable – Identify what key fields can be used to uniquely identify the selected fields. This can be a sequence variable, group ID, or unique date variable.
5. Unique Subject ID – Identify the variable which contains the unique subject identification value.

A graphical user interface can be used to assist in making these decisions.



The interface also has a “find related” tool which assists you in identifying fields that are potentially considered to be a related record field. It searches through the variable names and labels for key words. A report is then generated showing the possible related field.

Find Related Domain for: DEATH

Obs	Dataset and Variable	Data Table Name	Variable Name	Variable Length	Variable Label	Variable Type	Matched Criteria
1	death.event	death	EVENT	8	EVENT LEADING TO DEATH	N	Variable label (to)

Generated on: 02/12/2006, 5:03:10 pm, Sy Truong
 Located at: C:\GLOBAL\PROJECT1\STUDY1\SOURCE DATA

In this example, the key word it found was the word “to” in the label. This report is an example of how the tool can assist in expediting the selection and the creation of your related record domain dataset.

Comments

An analysis file or source data from an operational database usually has the comment fields stored in the same dataset upon which it is commenting. For example, if there is a comment captured on a CRF pertaining to adverse events, you would find this in the adverse event dataset. CDISC data is different in that all comments from all different sources are gathered together and stored separately in their own dataset named CO. In doing so, you have to identify additional information such as to which domain the comment is related, among other identification variables. The decision process is similar to SUPPQUAL and RELREC. Similar to the “find related”, there is a tool named “find comment”. This will search through variables and labels finding possible comment fields. This is usually pretty accurate since comment fields often have labels that have key words such as “comment” in them.

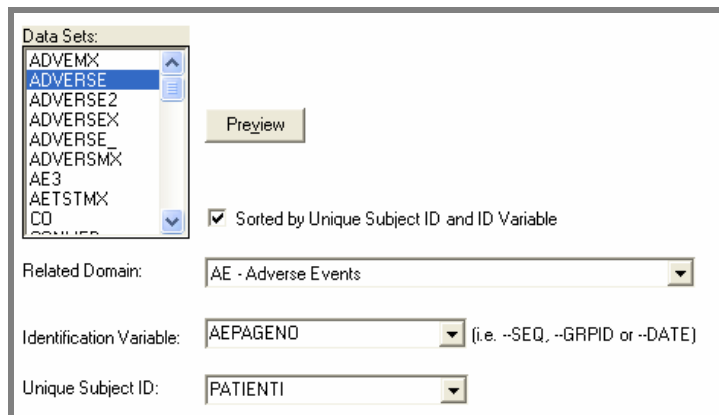
The three special purpose structures defined by CDISC are very flexible. They are vertical in structure so they can handle just about any source data. It is, however, unusual for data to be stored in this manner when being entered or analyzed. It is therefore necessary to perform transformations, which are time consuming since they must be handled one variable at a time. Automated macros and tools can help expedite these types of transformations.

STEP 8: SEQUENCE, ORDER AND LENGTHS – Data value sequence along with variable order and length need to also follow standards. CDISC does specify guidance for data sequences and variable order but is not as strict on variable lengths. In either case, you need to apply these consistently in order for the data to be standardized.

Sequence

Any dataset that contains more than one observation per subject requires a sequence variable. The sequence variable identifies the order of the values for each subject. If your data does not contain this sequence variable, you need to add it. Besides the subject ID, you also need to identify a unique identifier variable that distinguishes between the observations within one subject. This can be another group type of identification variable or a form date.

A tool named ADDSEQ adds this sequence based upon the choices you decide for a specific dataset.



The ADDSEQ tool will then create a new sequence variable containing sequential values after it sorts the data by the subject ID and identification variable. In addition to creating the sequence variables, there is also a tool that tells you if the dataset requires a sequence variable or not. It essentially verifies if there is more than one observation per subject. This will then help prompt you to add sequence variables in case that has been overlooked.

Variable Order

The data that is delivered in CDISC format needs to be ordered in a standard manner. All the key fields need to be first in order followed by the rest of the variables. The rest of the variables are then shown in alphabetical order. SAS datasets have their variables stored in a specified order and it is not necessarily in this standard order. A standard tool can re-order the variables with the keys appearing first, followed by the rest of the variables. The rest can be optionally alphabetized or left in their original order. This task may appear mundane but can be very helpful for the reviewer who is navigating through many datasets.

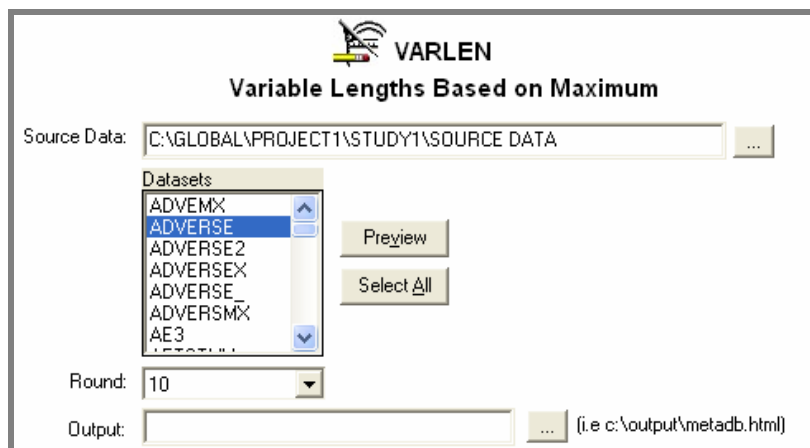
Variable Lengths

Variable lengths are not strictly specified by CDISC guidelines. Nevertheless, it is important to have variable lengths follow a standard for consistency. This includes:

1. Consistent lengths between variables that are the same across different data domains

2. Optimal lengths set to handle the data

In order to accomplish the first rule of standards, if you were to assign a length for one variable such as USUBJID in one dataset, you should set this variable to the same length in all other datasets. The second rule suggests that if the maximum length of this variable across all datasets is 9 characters, it would probably be optimal to set the length to 10. It makes sense to round up to the nearest ten to give it some buffer but not too much so that it would not be wasteful. Datasets can be very bloated and oversized for the values they carry if the second rule is not applied. The following tool named VARLEN assigns the length optimally.



In this example, the rounding option can be set to 10, 20 or none. It can therefore assign the exact maximum length which the data value contains if that is what is required. This creates the proper length statement so that your data will have the optimal lengths used for the values stored in that data.

STEP 9: DATA DEFINITION DOCUMENTATION – When you plan for a road trip, you need a map. This is analogous to understanding the data that is going to be part of an electronic submission. The reviewer requires a road map in order to understand what all the variables are and how they are derived. It is in the interest of all team members involved to have the most accurate and concise data documentation. This can help your team work internally while also speeding up the review process, which can really make or break an electronic submission to the FDA.

Levels of Metadata

There are several steps in documenting the data definition, most involving metadata, which is information about the data. There are several layers to the metadata. These include:

1. **General Information** – This pertains to information that affects the entire set of datasets. It could be things such as the name of the study, the company name, or location of the data.
2. **Data Table** – This information is at the SAS dataset level. This includes things such as the dataset name and label.
3. **Variable** – This information pertains to attributes of the variables within a dataset. This includes such information as variable name, label, and length.

The order in which the metadata is captured should follow the same order as the layers that are described.

Capture General Information

The following table lists the types of information that are important:

Metadata	Description
Company Name	This is the name of the organization that is submitting the data to the FDA.
Product Name	The name of the drug that is being submitted.
Protocol	The name of the study on which the analysis is being performed, which includes this set of data.

Layout	The company name, product name, and protocol are all going to be displayed on the final documentation. The layout information will describe if it will be in the footnote or title and how it is aligned.
--------	---

This high level metadata will be used in headers and footers on the final documentation.

Dataset Level Information

Some of the dataset level information can be captured through PROC CONTENTS but others need to be defined when you are documenting your data definition. Some of the information includes:

Metadata	Description
Data Library	Library name defines the physical path, server, and location of the data. This can also be in the form of a SAS LIBNAME.
Key Fields	Keys usually correlate to the sort order of the data. These variables are usually used to merge the datasets together.
Format Library	This is where the SAS format catalog is stored, if applicable.
Dataset Name	The name of each SAS dataset included.
Number of Variables	A count of the number of variables for each dataset.
Number of Records	Number of observations or rows within each dataset.
Dataset Comment	A descriptive text describing the dataset. This can contain the dataset label and other descriptive text explaining the data.

SAS Tools such as PROC CONTENTS can contribute to most of these items. However, comments and key fields can be edited which may differ from what is stored in the dataset.

Variable Level Information

The last step and level to the domain documentation is the variable level. This includes the following:

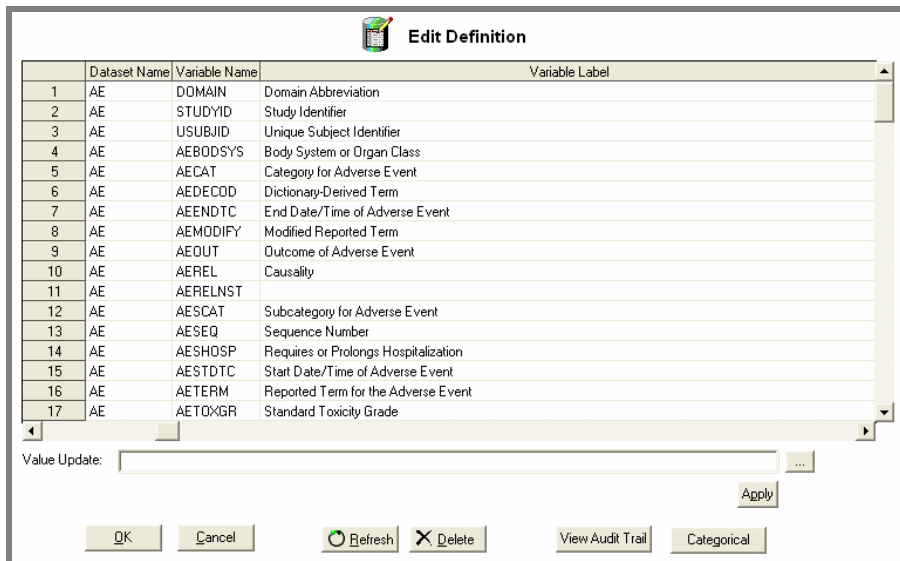
Metadata	Description
Variable Name	The name of the SAS variable.
Type	The variable type, such as Character or Numeric.
Length	The variable length.
Label	The descriptive label of the variable.
Format	SAS formats used. If it is a user defined format, it would need to be decoded.
Origins	The place from where the variable came. Sample values include Source and Derived.
Role	This defines the role of the variable. Example values include Key, Ad Hoc, Primary Safety, and Secondary Efficacy.
Comment	This is descriptive text explaining the meaning of the variable or how it was derived.

Similar to the data set level metadata, some of the variable level attributes can be captured through PROC CONTENTS. However, fields such as origins, role, and comments need to be edited by someone who understands the meaning of the data.

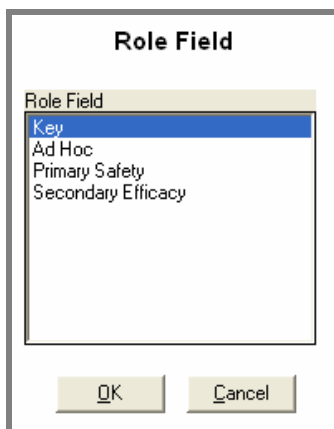
Automation

Tools such as PROC CONTENTS and Excel do have capabilities to customize and automate the documentation to a degree. They are not however intended specifically for creating data definition documentation. These tools therefore have limitations. A tool was developed entirely in SAS specifically for generating this type of documentation. This tool contains both a

graphical user interface and a macro interface to fit the user's requirements. The tool addresses all the disadvantages of the manual methods. It uses a similar PROC CONTENTS type of mechanism for capturing the initial metadata. However, it only retains the specific information that is pertinent to the data definition documentation.



Definedoc automatically captures attributes produced by PROC CONTENTS. For other attributes, it presents possible values that users can select for ease and consistency.



The tool also keeps track of all edits in an audit trail capturing who has updated what column so that if anything goes wrong, it can easily be traced back and fixed. One of the main advantages is that if any of the variable attributes are updated, this can be "refreshed" with a click of a button. It will not affect those fields that the user has entered, but rather, it updates other attributes such as variable names and labels.

Definedoc has the flexibility of exporting the pertinent information to an excel spreadsheet so that those users who prefer to edit their values within Excel can do so.

	A	B	C	D	E	F	G	H	I	J	K
1	Library	Dataset	Variable	Type	Length	Variable Label	Format	Decode	Origins	Role	Comment
2	c:\apache\AE	DOMAIN	Character	30	Domain Abbreviation				Derived		
3	c:\apache\AE	STUDYID	Character	10	Study Identifier				Source	Key	
4	c:\apache\AE	USUBJID	Character	10	Unique Subject Identifi	11			Source	Key	5-DIGIT SUBJECT IDENTIF
5	c:\apache\AE	AEBODS	Character	30	Body System or Organ Class				Source		
6	c:\apache\AE	AECAT	Character	30	Category for Adverse Event				Source		VALUES: ADEERS, DOSE
7	c:\apache\AE	AEDECOT	Character	40	Dictionary-Derived Term				Source		
8	c:\apache\AE	AEENDTC	Character	10	End Date/Time of Adverse Event				Source	Key	
9	c:\apache\AE	AEMODIF	Character	40	Modified Reported Term				Source		
10	c:\apache\AE	AEOUT	Character	1000	Outcome of Adverse Event				Source		AE DESCRIPTION WAS SF
11	c:\apache\AE	AEREL	Character	260	Causality				Source		
12	c:\apache\AE	AERELNS	Character	195					Source		CONCOMITANT MEDICAT
13	c:\apache\AE	AESCAT	Character	30	Subcategory for Adverse Event				Source		
14	c:\apache\AE	AESEQ	Numeric	8	Sequence Number				Derived		
15	c:\apache\AE	AESHOSF	Character	30	Requires or Prolongs Hospitalization				Source		VALUES: NO, YES.
16	c:\apache\AE	AESTDTC	Character	10	Start Date/Time of Adverse Event				Source		
17	c:\apache\AE	AETERM	Character	150	Reported Term for the Adverse Event				Source		
18	c:\apache\AE	AETOXGF	Character	10	Standard Toxicity Grade				Source		VALUES: 1, 2, 3, 4, 5.
19	c:\apache\CM	DOMAIN	Character	30	Domain Abbreviation				Derived		
20	c:\apache\CM	STUDYID	Character	10	Study Identifier				Source	Key	

This provides the best of both worlds. It captures just the values that you want and exports this to Excel for those who prefer this interface. Once you are finished with editing the information in Excel, the same spreadsheet can be re-imported so that the information is handled centrally. Besides the dataset and variable level metadata information, Definedoc also helps automate the capture of the high level general information.

General Information

Output Library:

Company Name:

Product Name:

Protocol:

XPT Link Location: ... (*)
(i.e. .../xptfiles)

CRF Link Location: ... (*)

Split Char:

Layout

Left Title: Center Title: Right Title:

Left Footnote: Center Footnote: Right Footnote:

* NOTE: This is relative to the output path: c:\temp.

This handles both the editing of the information and layout of the final report.

Generating Documentation

The last step in the process is to generate the documentation in either PDF or XML format. The challenge is that in order to make the documentation useful, it requires hyperlinks to link the information together. The manual method allows you to format the information in Word and this can be converted into PDF. Even though Word and Excel can generate XML, they do not have the proper schema so there is no manual way of generating the needed XML version of the report. The definedoc has the flexibility of generating the report in Excel, RTF, PDF, and XML.

Output File: ...

(i.e. define.xls, define.rtf, define.pdf, define.xml)

It utilizes ODS within SAS to produce output in all these formats. In addition to the XML file, the definedoc tool also produces the accompanying cascading style sheet to format the XML so that you can view this within a browser in a similar format as in a web browser. An example PDF output would look like this:

Variable Name	Type	Length	Variable Label	Format	Origins	Role	Comment
DOMAIN	Character	30	Domain Abbreviation		Derived		
STUDYID	Character	10	Study Identifier		Source	Key	
USUBJID	Character	10	Unique Subject Identifier	11.	Source	Key	5-DIGIT SUBJECT IDENTIFIER (NUMERIC)
AEBODSYS	Character	30	Body System or Organ Class		Source		
AECAT	Character	30	Category for Adverse Event		Source		VALUES: ADEERS, DOSE DISCONTINUATION FORM 2112, DOSE REDUCTION FORM 2112, ON-STUDY (EPP), ON-STUDY FORM 1560 (NON-EPP), TOXICITY (EPP), TOXICITY FORM 1560 (NON-EPP).
AEDECOD	Character	40	Dictionary-Derived Term		Source		
AEENDTC	Character	10	End Date/Time of Adverse Event		Source	Key	
AEMODIFY	Character	40	Modified Reported Term		Source		
AEOUT	Character	1000	Outcome of Adverse Event		Source		AE DESCRIPTION WAS SPLIT INTO 10 VARIABLES TO COMPLY WITH LENGTH LIMITATIONS FOR SAS TRANSPORT FILES. CONCATENATE FIELDS 1-10 TO RECREATE THE ORIGINAL TEXT.
AEREL	Character	200	Causality		Source		
AERELNST	Character	195			Source		CONCOMITANT MEDICATION ATTRIBUTION AND NAME.

CONCLUSION

There are many challenges in working with CDISC SDTM version 3.1.1. It can accommodate a wide variety of data but it is structured differently from data on which users would perform analysis. CDISC is intended to be used for submissions, so transformation will likely be necessary. Since transformations are different for each variable, the sum of the work can be tremendous. It requires organization before execution and optimization in implementation. The techniques, methodologies, and tools presented in this paper demonstrate ways of optimizing conversion to CDISC structures based on real world experience. Armed with efficient and tested approaches, you can avoid mistakes and implement CDISC with success.

REFERENCES

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

CDISC Builder, Transdata and Definedoc and other MXI (Meta-Xceed, Inc.) product names are registered trademarks of Meta-Xceed, Inc. in the USA.

Other brand and product names are registered trademarks or trademarks of their respective companies.

ABOUT THE AUTHOR

Sy Truong
 Meta-Xceed, Inc.
 1751 McCarthy Blvd.
 Milpitas, CA 95035
 (408) 955-9333
 sy.truong@meta-x.com

Patricia Gerend
 Genentech, Inc.
 1 DNA Way
 South San Francisco, CA 94080
 (650) 225-6005
 gerend@gene.com