Paper CD12

Discover Define-XML

Mark Wheeldon, Formedix, Burlington, MA, US
Kevin Burges, Formedix, Burlington, MA, US

ABSTRACT

Historically, Define.pdf was used as a submission deliverable to enable FDA reviewers to understand the content of submitted proprietary datasets. Define-XML was introduced by CDISC as a machine-readable alternative, and this has replaced Define.pdf as a mandatory part of an eCTD-based FDA submission. Define-XML is still often seen as an afterthought, only produced when datasets are ready to be submitted.

INTRODUCTION

Since its inception in 2000, the Clinical Data Interchange Standards Consortium (CDISC) has developed and supported globally-adopted data standards to improve clinical trial efficiency. Clinical data standards are now recognized as playing a vital role in the entire end-to-end clinical trial process, as well as introducing operational and time savings in the development of new drugs.

One of the most widely used standards today is Define-XML. Based on a recent poll[1], 71% of those currently using CDISC have implemented Define-XML.

This paper explores alternative uses of Define-XML from study set-up through the production of CDISC Study Data Tabulation Model (SDTM) datasets in study conduct and analysis and onto submission. In addition, the new capabilities of Define-XML 2.0 will be explored.

DEFINE-XML: THE MYTH AND THE REALITY

It is commonly thought that Define-XML is simply a way to document what datasets look like - the names and labels of datasets and variables, what terminology is used etc. Define-XML does this in great detail but thinking of this dataset metadata purely as a dataset descriptor circumvents its true potential.

Progressive uses of Define-XML – which have primarily focused on using it to design datasets upfront, alongside CRFs - have seen Define-XML optimize the end-to-end clinical trial process in the following ways:

- Establish Libraries and Templates
    - Re-use of all dataset designs (SDTM, proprietary EDC, Analysis (ADaM)) end-to-end
    - Storage of any type of dataset design
    - Standards governance of dataset designs
    - Dataset mappings and presentation of Define.pdf can also be reused
- Study Set-up and Build
    - Streamline design from protocol to submission
    - Findings, interventions and events core variables facilitate design of new CRF content.
    - Configuration of data transformation tools using machine-readable Define-XML
- Study Conduct and Analysis
    - Define-XML metadata driven dataset conversions
    - Initiation and automation of clinical data warehouse load
    - Automated validation of study designs versus standards and delivered datasets against upfront dataset designs

- Submission
    - Auto-generation of human-readable Define.pdf/Define.html with table of contents, bookmarking and annotated CRF hyperlinks

STREAMLINING STUDY SET-UP AND CRF DESIGN

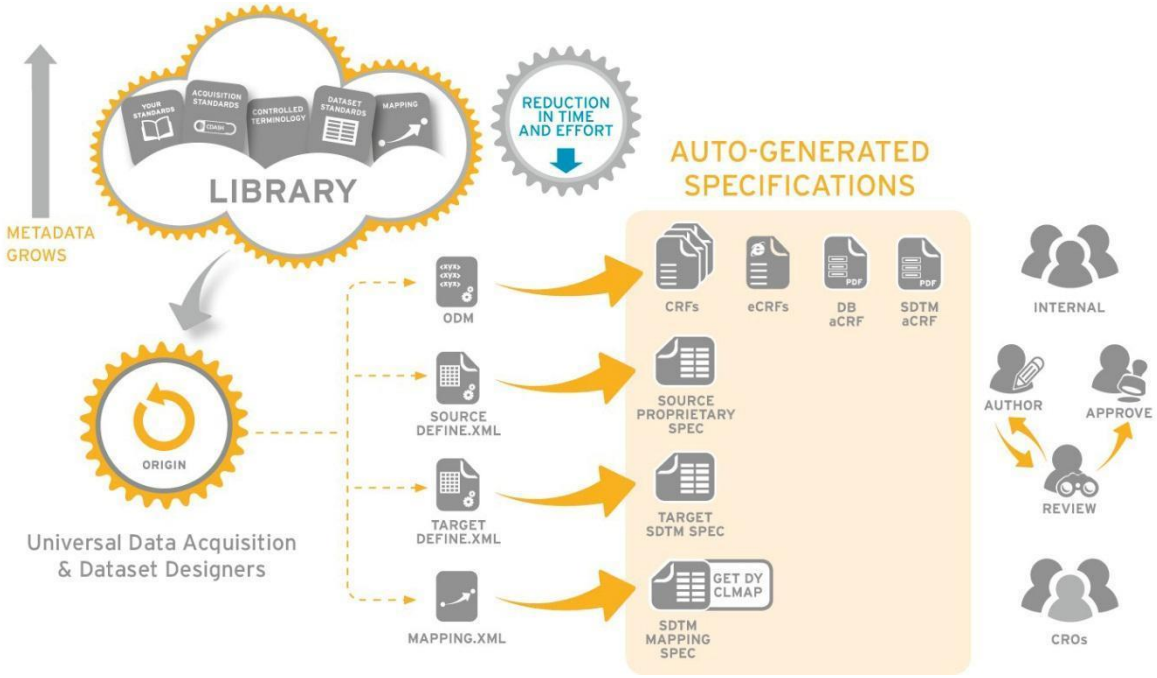


Figure 1. Streamlining study set-up and CRF design

Designing submission datasets using Define-XML and SDTM at the start of a study (see Figure 1, "Target Define.xml/Target SDTM spec") aids form creation and makes it easy to check that CRFs contain all the information required to generate the datasets.

SDTM gives all Identifier, Topic, Qualifier and Timing variables, which can all be used when designing the form (see Figure 1, "ODM/CRF and eCRFs").

SDTM datasets designed upfront aid dataset annotation of CRFs with SDTM variables (see Figure 2 below). This has the additional benefit of providing basic mapping between the forms and the datasets. CDISC provides a mechanism to extend Define-XML which is permissible and allows the storage of additional metadata such as complex dataset mappings (e.g. how data may be merged into one single dataset from two sources). We call this "mapping.xml" which will be discussed in more detail later.

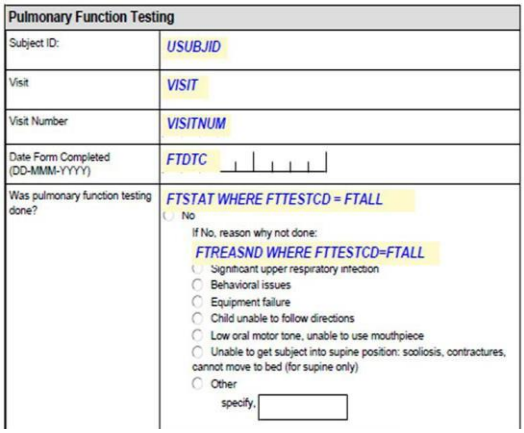


FIGURE 2. SDTM ANNOTATED CRF

FACILITATING EDC DATA CONVERSIONS

Define-XML is, in fact, not just limited to describing CDISC SDTM and ADaM dataset structures. EDC systems export their own proprietary dataset formats which can be described using the Define-XML model. With the right tools, a Define-XML describing the EDC export datasets can be automatically generated from CRFs/eCRFs (see Figure 1, "Source Define.xml"). This can then be displayed in a friendly HTML or PDF format (Figure 1, "Source Proprietary Spec") allowing visibility, at the start of a study, of the datasets that will be delivered by the EDC system.

The Source Proprietary dataset spec enables upfront mapping of EDC datasets to SDTM datasets. These mappings can be described (and made machine-executable) using "mapping.xml" and human-readable SDTM mapping specifications can be produced automatically: aiding review and approval of mappings.

In addition, mapping.xml provides a machine-executable format which can be processed by data transformation code to enable the automatic conversion of datasets in SAS®, Informatica® or other commercially available tools.

Mapping.xml is not just limited to mapping EDC to SDTM datasets: it can also be used to convert data in the CDISC Operational Data Model (ODM) to SDTM, and also to convert SDTM datasets into data marts for loading into a clinical data warehouse, for example.

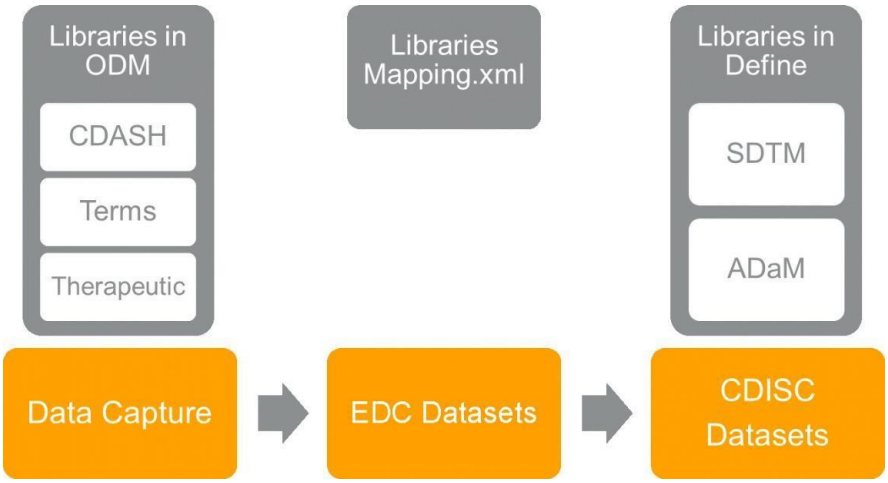


Figure 3. Data flows from data capture to datasets and how CDISC metadata fits in.

The diagram above shows the flow of data from data capture and onto CDISC datasets and the part CDISC metadata plays (in designing data capture forms using CDISC ODM and Define-XML in designing destination datasets). All of this vendor neutral metadata can form the basis of form and dataset libraries which can be reused from study to study.

CREATING AND REUSING DATASET LIBRARIES

Define-XML is the perfect vehicle to store libraries of datasets (EDC, SDTM, ADaM, other), mappings, page links to CRF variables, etc., for reuse from one study to the next.

We have illustrated that a metadata driven approach using Define-XML can optimize a single study from set-up to submission, but creating libraries of metadata that can be re-used will make future studies even more efficient.

Libraries of data acquisition forms, proprietary EDC datasets, SDTM datasets, ADaM and dataset mappings ensure that only new content has to be created for each study and that only these new objects must be validated from study to study.

Figure 4 below shows the effect of reuse on only one part of the end-to-end clinical trial – database build[2] - but shows that after building a library based on 5 studies, around 70% of metadata for a study can be re-used from the library, rising to around 85% after 35 studies.

Our clients[3] have reported 70% reuse end-to-end using libraries in this manner.

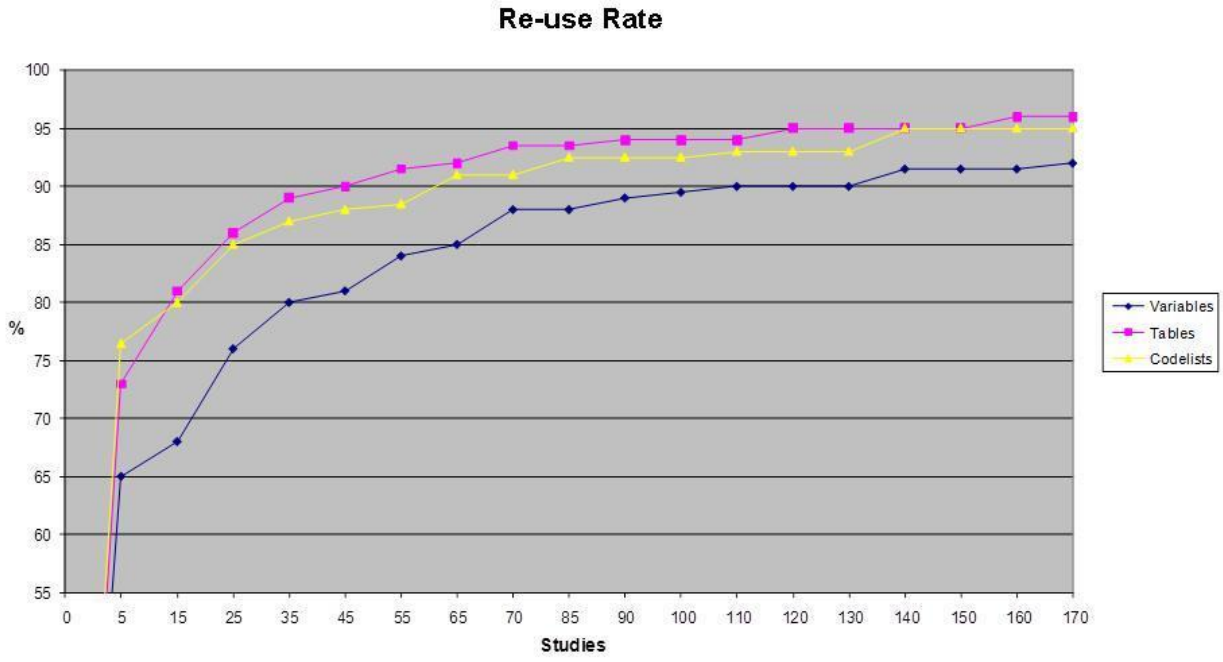


Figure 4. Standardization and reuse.

AUTOMATING DATASET VALIDATION

Defining datasets upfront provides another major advantage. By moving to prospective definition of the intended datasets, it is then possible to machine validate study dataset designs for conformance to external standards. It is also possible to validate that data populated datasets match the original specifications. Data quality and submission compliance are built in upfront, with less reliance on downstream validation.

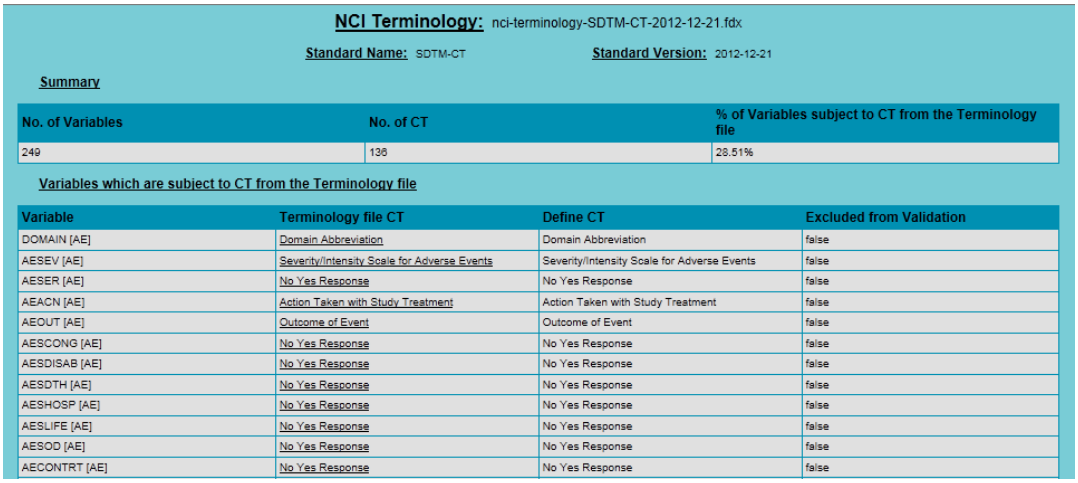
Now it is possible to automatically perform the following validation tasks:

1. Comparing Study Dataset Designs and Controlled Terminology to External and Internal standards

   First and foremost, when designing SDTM datasets and creating controlled terms, it is imperative that these comply with the latest and/or chosen version of SDTM or National Cancer Institute Controlled Terminology (NCI CT). During the dataset design phase, automatic comparisons and compliance checks can be made with the appropriate version of CDISC SDTM, ADaM and NCI-CT.

In addition, companies are moving towards developing their own domains which comply with CDISC SDTM, but with content which falls outside of the standard safety domains (for example, specialist findings domains may be required for a particular therapeutic area). The same situation also occurs for controlled terms as the NCI-CT coverage of controlled terminology is still quite small. Again, companies can compare study dataset designs against their own data standards (stored in Define-XML) to check for differences and either accept or reject them accordingly.

See Figure 5, which shows an example of a report of NCI CT coverage.



NCI Terminology: nci-terminology-SDTM-CT-2012-12-21.fdx

Standard Name: SDTM-CT **Standard Version:** 2012-12-21

Summary

No. of Variables	No. of CT	% of Variables subject to CT from the Terminology file
249	136	28.51%

Variables which are subject to CT from the Terminology file

Variable	Terminology file CT	Define CT	Excluded from Validation
DOMAIN [AE]	Domain Abbreviation	Domain Abbreviation	false
AESEV [AE]	Severity/Intensity Scale for Adverse Events	Severity/Intensity Scale for Adverse Events	false
AESER [AE]	No Yes Response	No Yes Response	false
AEACN [AE]	Action Taken with Study Treatment	Action Taken with Study Treatment	false
AEOUT [AE]	Outcome of Event	Outcome of Event	false
AESCONG [AE]	No Yes Response	No Yes Response	false
AESDISAB [AE]	No Yes Response	No Yes Response	false
AESDTH [AE]	No Yes Response	No Yes Response	false
AESHOSP [AE]	No Yes Response	No Yes Response	false
AESLIFE [AE]	No Yes Response	No Yes Response	false
AESOD [AE]	No Yes Response	No Yes Response	false
AECONTRT [AE]	No Yes Response	No Yes Response	false

Figure 5. NCI CT coverage

2. Comparing “As Specified” Study Dataset Specifications against “As Delivered” Study Dataset Designs

   Increasing numbers of studies are outsourced to Contract Research Organizations (CROs) and this leads to an increased sponsor burden in two areas: a) upfront specification of deliverables and b) downstream validation of those deliverables.

   A machine-executable target SDTM Define-XML specification and a corresponding human-readable target SDTM specification (in HTML, PDF, Word or Excel) would be given to a CRO to describe what would be expected in delivered datasets.

   Now, when CROs return the datasets, they should also provide study dataset metadata (an “as delivered” Define-XML). With both “as specified” and “as delivered” study dataset metadata available, it is easy to compare the study dataset metadata to verify that the “as delivered” dataset actually matches what was specified.

3. Comparing Dataset Data to Dataset Metadata

   Again, having upfront target SDTM Define-XML available allows automated comparison of delivered datasets against study dataset metadata, either “as specified” or “as delivered”. Comparing data to “as specified” Define-XML verifies that the data matches what was originally intended/specified, while comparing data to “as delivered” Define-XML ensures that the data matches the dataset definition. This is important as it will ultimately be this “as delivered” Define-XML that will be submitted to the FDA.

DEFINE-XML 2.0: WHAT'S CHANGED?

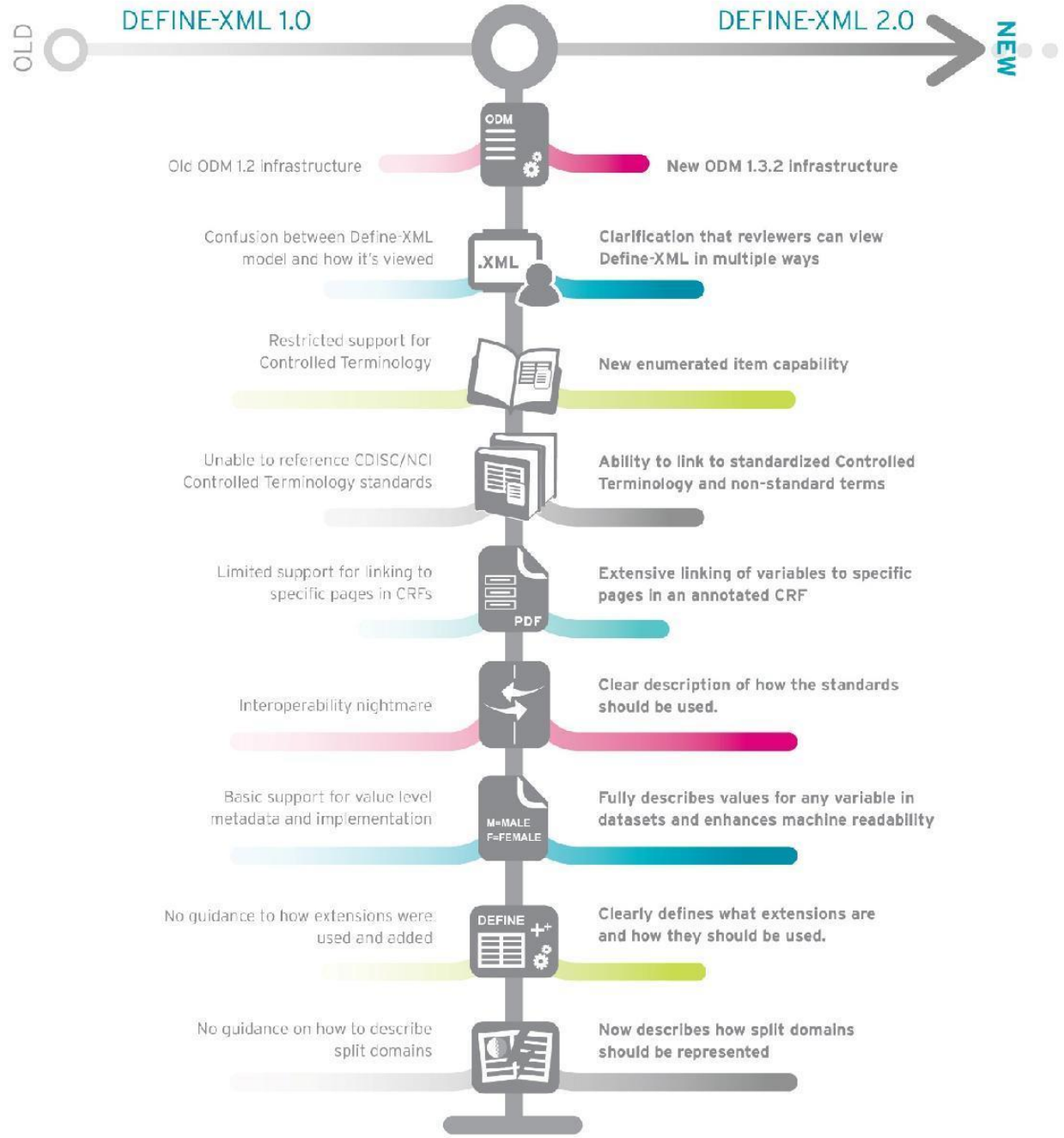


Figure 6. Changes in Define-XML 2.0

Define-XML 2.0 introduces many improvements over the original, which are summarized in Figure 6. These changes are focused around removing ambiguity, improving value level metadata, linking to CDISC/NCI Controlled Terminology standards and linking to annotated CRFs.

VALUE LEVEL METADATA RE-IMAGINED

Value Level Metadata is needed where values in a table column may have different metadata depending on the row they are in. For example, the content of VSORRES might be DataType=”integer” for one value of VSTESTCD, but DataType=”float” for another.

Define-XML 1.0 supports this by providing a Value List that defines the content of VSORRES for each test code. It does this “by convention”, however, with no clear, unambiguous way to know exactly what the Value List is defining. It also doesn’t describe how Value Lists can be used to provide Value Level Metadata for multiple columns, for example, where VSORRES and VSPOS both have different definitions, depending on the test code.

Different organizations interpreted the specification in different ways and ended up with incompatible implementations. Define-XML 2.0 removes this ambiguity by allowing Value Lists to be provided explicitly for each variable in the dataset. This allows full description of the metadata for any value in any variable.

WHERE CLAUSES

There is a new mechanism to describe the conditions under which each value definition is applicable. Where Clauses define a condition, such as “Where VSTESTCD=SYSBP”, these conditions are linked to the Value Level Metadata so that it is unambiguously known when each definition applies. Compound conditions can be used, such as “Where VSTESTCD=SYSBP and VSPOS=SITTING” (see Figure 7).

```
<!-- Where Clause definitions for: Where VSTESTCD = 'SYSBP' and VSPOS = 'SITTING' -->
<def:WhereClauseDef OID="WC.VS.VSTESTCD.SYSBP.VS.VSPOS.SITTING">
  <RangeCheck SoftHard="Soft" def:ItemOID="IT.VS.VSTESTCD" Comparator="EQ">
    <CheckValue>SYSBP</CheckValue>
  </RangeCheck>
  <RangeCheck SoftHard="Soft" def:ItemOID="IT.VS.VSPOS" Comparator="EQ">
    <CheckValue>SITTING</CheckValue>
  </RangeCheck>
</def:WhereClauseDef>
```

Figure 7. Compound Where Clause in Define-XML 2.0

These were the most-requested features for Define-XML 2.0. They also provide support for ADaM parameter-level metadata.

SLICES

The Value List mechanism is focused around variables, grouping together the definitions of all the possible values for a given variable. Using the information provided by the new Where Clause mechanism, Value Level Metadata can be displayed in a transposed format that shows how a particular “Slice” of a dataset looks. Rather than seeing how a specific variable looks for each condition, a Slice shows how the whole dataset looks for a given condition.

Slices and Value Lists are simply two different ways of looking at the same underlying metadata. Figure 8 shows Value Level Metadata represented as Value Lists, specifically an example set of Values for the VSORRES and VSORRESU Variables.

Value List for Vital Signs (VS) Domain VL1

Source Variable	Label	Type	Controlled Terms or Format	Origin	Role	Comment
VSORRES	Temperature	float				*WHERE VSTESTCD EQ TEMP and COUNTRY IN UK, AUSTRALIA* *WHERE VSTESTCD EQ TEMP and COUNTRY IN USA*
VSORRES	Heart Rate	integer				*WHERE VSTESTCD EQ HR*

Value List for Vital Signs (VS) Domain VL2

Source Variable	Label	Type	Controlled Terms or Format	Origin	Role	Comment
VSORRESU	Temperature	text	["C"]			*WHERE VSTESTCD EQ TEMP and COUNTRY IN UK, AUSTRALIA*
VSORRESU	Temperature	text	["F"]			*WHERE VSTESTCD EQ TEMP and COUNTRY IN USA*
VSORRESU	Heart Rate	text	["BPM"]			*WHERE VSTESTCD EQ HR*

Figure 8. Value Level Metadata represented as Value Lists

Viewing this metadata as a Slice presents what the entire Domain looks like for a given condition, as shown in Figure 9.

Slice: VS Domain WHERE VSTESTCD EQ TEMP and COUNTRY IN UK, AUSTRALIA

Name	Label	Type	Controlled Terms	Origin	Role	Comment
STUDYID	Study Identifier	text				
DOMAIN	Domain Abbreviation	text	["VS"]			
USUBJID	Unique Subject Identifier	text				
VSSEQ	Sequence Number	float				
VSTESTCD	Vital Signs Test Short Name	text	Vital Signs Test Code (C66741)			
VSTEST	Vital Signs Test Name	text	Vital Signs Test Name (C67153)			
VSORRES	Temperature	float				
VSORRESU	Temperature	text	["C"]			
VSSTRESC	Character Result/Finding in Std Format	text				
VSSTRESN	Numeric Result/Finding in Standard Units	float				
VSSTRESU	Standard Units	text	Units for Vital Signs Results (C66770)			
VSBLFL	Baseline Flag	text	No Yes Response (C66742)			
VISITNUM	Visit Number	float				
VSDTC	Date/Time of Measurements	datetime	ISO 8601 (Dates/Times)			

Figure 9. Value Level Metadata represented as a Slice

ENUMERATED ITEMS

In Define-XML 1.0, Controlled Terminology definitions had to have both a coded value and a decode given. In most cases, these were the same as SDTM, which uses controlled lists of values rather than having codes and decodes. Define-XML 2.0 leverages the new enumerated items mechanism in ODM 1.3.1, so Controlled Terminology can be defined simply as a list of allowable values if there is no code/decode relationship. Figure10 shows how this is used to describe a Severity Code List.

```
<CodeList OID="CL.SEV" Name="Severity" DataType="text">
  <EnumeratedItem CodedValue="MILD"/>
  <EnumeratedItem CodedValue="MODERATE"/>
  <EnumeratedItem CodedValue="SEVERE"/>
</CodeList>
```

Figure 10. Use of Enumerated Items

STANDARDIZED CONTROLLED TERMINOLOGY

Define-XML 2.0 allows linking of code lists, and even individual codes, to the published CDISC and NCI Controlled Terminology standards using Aliases. It also allows codes to be flagged as “extended values”, where a sponsor has added additional codes to an extensible code list from those Controlled Terminology standards.

Figure 11 shows how this is used to reference the SCTESTCD codes, both at the Code List and the Code List Item level, by adding an Alias that points at the standard “C” codes.

```
<CodeList OID="CL.SCTESTCD" Name="Subject Characteristic Code (SCTESTCD)" DataType="text"
SASFormatName="$SCTESTC">
  <CodeListItem CodedValue="EDLEVEL">
    <Decode>
      <TranslatedText xml:lang="en">Education Level</TranslatedText>
    </Decode>
    <Alias Name="C17953" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <CodeListItem CodedValue="MARISTAT">
    <Decode>
      <TranslatedText xml:lang="en">Marital Status</TranslatedText>
    </Decode>
    <Alias Name="C25188" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <CodeListItem CodedValue="SUBJINIT" def:ExtendedValue="Yes">
    <Decode>
      <TranslatedText xml:lang="en">Subject Initials</TranslatedText>
    </Decode>
  </CodeListItem>
  <Alias Name="C74559" Context="nci:ExtCodeID"/>
</CodeList>
```

Figure 11. Referencing NCI Controlled Terminology

ENHANCED DATA TYPES AND DATA TYPE GUIDANCE

The latest release of Define-XML introduces a richer set of data types, and defines how these should be used in relation to the SAS Char and Num data types that are used in SDTM. This allows for better specification of the expected data and, as a result, the possibility of better checking of the data against those data types.

ENHANCED LINKING

Define-XML 2.0 introduces the ability to link to a specific page or pages in a document. This is used in several places:

- Formedix Origin can now link to the specific page in an annotated CRF that a variable was collected on, as shown in Figure 12
- Comments can now link to sections in supplemental documents that provide information about variables
- Methods can now link to sections in supplemental documents that describe the derivation of a value, as shown in Figure 13.

```
<!-- Item Definition: Variable Level (BRTHDTC) -->
<ItemDef OID="IT.DM.BRTHDTC" Name="BRTHDTC" DataType="date" SASFieldName="BRTHDTC">
  <Description>
    <TranslatedText xml:lang="en">Date/Time of Birth</TranslatedText>
  </Description>
  <def:Origin Type="CRF">
    <def:DocumentRef leafID="LF.blankcrf">
      <def:PDFPageRef PageRefs="6" Type="PhysicalRef"/>
    </def:DocumentRef>
  </def:Origin>
</ItemDef>
```

Figure 12. Linking to pages in an annotated CRF

```
<!-- Method Definition: Algorithm included or expanded in an external file -->
<MethodDef OID="MT.EGDRVFL" Name="Algorithm to derive EGDRVFL" Type="Computation">
  <Description>
    <TranslatedText xml:lang="en">EGDRVFL = "Y" for derived EGTESTCDs QTCB and QTCF. Null otherwise.
    </TranslatedText>
  </Description>
  <def:DocumentRef leafID="LF.ComplexAlgorithms">
    <def:PDFPageRef PageRefs="EG" Type="NamedDestination"/>
  </def:DocumentRef>
</MethodDef>

<def:leaf ID="LF.ComplexAlgorithms" xlink:href="complexalgorithms.pdf">
  <def:title>Complex Algorithms</def:title>
</def:leaf>
```

Figure 13. Linking to sections in a supplemental document

A similar mechanism can be used to link ADaM variables to their predecessors, as shown in Figure 14.

```
<!-- Item Definition: Variable Level (TRTP) -->
<ItemDef OID="IT.ADQSADAS.TRTP" Name="TRTP" DataType="text" Length="20">
  <Description>
    <TranslatedText xml:lang="en">Planned Treatment</TranslatedText>
  </Description>
  <CodeListRef CodeListOID="CL.ARM"/>
  <def:Origin Type="Predecessor">
    <Description>
      <TranslatedText xml:lang="en">ADSL.TRT01P</TranslatedText>
    </Description>
  </def:Origin>
</ItemDef>
```

Figure 14. Linking to ADaM predecessor variables

ENHANCED DERIVATIONS

The old Define-XML 1.0 method of specifying derivations has been replaced by an improved implementation taken from the ODM 1.3 standard. This enables a variable that appears in multiple datasets to have a different derivation for each dataset it is present in.

CLARIFICATION OF HOW TO SPECIFY SPLIT DOMAINS

Define-XML 1.0 was released before split domains were introduced in SDTM 1.2, and so it does not define how the various properties of a domain should be used to specify both the core domain code (e.g., “QS”) and the extended split domain code (e.g., “QSCG”). Define-XML 2.0 now properly defines how this information should be specified.

CLARIFICATION OF EXTENDING DEFINE-XML

Due to ambiguity in Define-XML 1.0, there are varying opinions on what the core Define-XML model includes, whether other parts of the underlying ODM model can be used, and what extensions to the model can be used for. Define-XML 2.0 states clearly that:

- Anything not defined in the specification is considered an extension, even if it is part of the underlying ODM model
- Use of extensions from the underlying ODM model is not prohibited. However, they have no meaning with regard to the standard: their meaning must be agreed between the sender and receiver of the metadata
- Extensions that duplicate functionality in the core Define-XML 2.0 model are not allowed - this is to ensure all users apply the same mechanism for all functionality defined in the model
- Extensions cannot fundamentally change the meaning of the model: i.e. if all extensions are removed, the metadata essentially must have the same meaning as it did with the extensions present.

These clarifications were added to prevent fragmented implementations of Define-XML and, as such, allow applications to be confident of the meaning of a piece of Define-XML metadata.

DEFINING A MODEL, NOT A VIEW

The Define-XML 1.0 specification was intended to define the model that describes a set of datasets; however, due to the way it was presented, it was commonly interpreted to define how dataset metadata should be displayed for viewing.

Define-XML 2.0 tries to make it clear that it is the model that is being defined, not how it should be displayed. Define-XML 2.0 includes a stylesheet that demonstrates how the dataset metadata can be displayed, however, this display format is not part of the standard and implementers are free to display the dataset metadata in any way that is suitable for the receiver.

COMPATIBILITY

Define-XML 2.0 is based on and largely similar to the Define-XML 1.0 model, however, it is not completely backward compatible with it. Compatibility has been sacrificed in order to produce a cleaner, less ambiguous model. For example, using a value list on the –TESTCD variable to define the contents of the –ORRES or other variable is no longer permitted; a value list now always describes the variable that references it. To provide Value Level Metadata for multiple variables, simply attach a value list to each. Existing Define-XML 1.0 files will require updating to make them Define-XML 2.0 compliant. This updating process is fairly simple and can be automated.

Due to the ambiguities in Define-XML 1.0, it would not be possible to provide a single upgrade routine that would correctly upgrade all files from all systems, so it is left to system implementers to provide upgrade routines for their implementation of Define-XML 1.0 if they feel this is required.

CONCLUSION

Define-XML should not be regarded merely as a submission deliverable, but as a CDISC model that helps to optimize the whole end-to-end clinical trial process. It can be used to establish dataset libraries which promote study-to-study re-use, as well as driving efficiencies through expedited study set-up and streamlined dataset conversions in study conduct and analysis. Using Define-XML at the start of new study design makes it possible to machine-validate dataset deliverables, guaranteeing that data quality and submission compliance are built-in with less reliance on downstream validation.

Define-XML 2.0 provides a substantially enhanced and more robust mechanism for describing dataset metadata by allowing full specification of value and parameter level metadata for any variable, and improves interoperability and machine readability by removing ambiguity. This will lead to increased opportunities for automation and as such, drive further efficiencies in the study process.

REFERENCES

[1]Mark Wheeldon, Kevin Burges "Discover Define-XML, Practical Users Today.", March 14, 2013, www.formedix.com/.

[2]Schering-Plough, "Effect of Standards Libraries and Re-use in CDMS design", Dublin, May 2003.

[3] Stephane Auger, Danone Research, September 2013

[4]CDISC. "Define-XML 2.0". 2013-03-05. Available at http://cdisc.org/define-xml.

[5]CDISC. "ODM". Available at http://www.cdisc.org/odm

ACKNOWLEDGMENTS

The authors would like to acknowledge the help of all of the Formedix team in preparing this paper and clients who use Define-XML upfront in their studies today.

RECOMMENDED READING

- CDISC Define-XML Specification and Implementation Guides
- CDISC SDTM Standard and Implementation Guide.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name: Mark Wheeldon
Enterprise: Formedix
Address: 15 New England Executive Park
City, State ZIP: Burlington, MA, 01803
Work Phone: +1 781 791 9506
E-mail: markwheeldon@formedix.com
Web: www.formedix.com
Twitter: @formedixinc

Co-author: Kevin Burges
Address: As above
E-mail: kevinburges@formedix.com