Case Report Tabulation Data Definition Specification (define.xml)

Prepared by the
CDISC define.xml Team

Principal Editor: William Qubeck
Principal Contributors: Sally Cassells, Anthony Friebel, and the define.xml team

Notes to Readers

This version of the Case Report Tabulation Data Definition Specification supersedes all prior versions. Version 1.0.0 reflects changes from a comment period through the Health Level 7 (HL7) Regulated Clinical Research Information Management Technical Committee (RCRIM) in December 2003 (www.hl7.org) and CDISC's website in September 2004 as well as the work done by the define.xml team to add functionality, features, and additional documentation.

Version 1.0.0 incorporated the applicable comments, suggestions, and corrections received from the two comment periods specified above and is the official implementation version.

Revision History

Date	Version	Summary of Changes	Primary Authors
2005-02-05	1.0.0	This is the official implementation version of the Case Report Tabulation Data Definition specification.	The define.xml team
2005-02-09	1.0.0	Administrative update.	Anthony Friebel, William Qubeck, Sally Cassells

Table of Contents

1. Introduction

1.1. Purpose of this document

This document specifies the standard for providing Case Report Tabulations Data Definitions in an XML format for submission to a regulatory authority such as the U.S. Food and Drug Administration (FDA). This specification is commonly referred to as "define.xml". The XML schema used to define the expected structure for these XML files is based on an extension to the CDISC Operational Data Model (ODM). The 1999 FDA electronic submission (eSub) guidance and the electronic Common Technical Document (eCTD) documents specify that a document describing the content and structure of the included data should be provided within a submission. This document is known as the Data Definition Document (e.g., "define.pdf" in the 1999 guidance). The Data Definition Document provides a list of the datasets included in the submission along with a detailed description of the contents of each dataset. To increase the level of automation and improve the efficiency of the Regulatory Review process, define.xml can be used to provide the Data Definition Document in a machine-readable format. The formal name for this is the Case Report Tabulation Data Definition (CRT DD) specification.

This specification defines the metadata structures that are to be used to describe the Case Report Tabulation datasets and variables in a manner that meets or exceeds the minimum FDA requirements.

1.2. U.S. Electronic Submission Background

In the United States, the approval process for regulated human and animal health products requires the submission of data from clinical trials and other studies as expressed in the Code of Federal Regulations (CFR). The FDA established the regulatory basis for wholly electronic submission of data in 1997 with the publication of regulations on the use of electronic records in place of paper records (21 CFR Part 11). In 1999, the FDA standardized the submission of clinical and non-clinical data and metadata in SAS Version 5 Transport (XPT) and Portable File Format (PDF), respectively. As specified within the 1999 FDA eSub guidelines, the metadata descriptions of the datasets (e.g., the content and structure) are within the Data Definition Document (define.pdf).

In 2003, the FDA published a set of guidance documents on receiving electronic product applications per the International Conference on Harmonization (ICH) electronic Common Technical Document (eCTD) specifications. Within these new specifications, the FDA expanded the acceptable file types to include XML.

1.3. CDISC

The Clinical Data Interchange Standards Consortium (CDISC) is a non-profit organization whose mission is to develop industry standards to support acquisition, exchange, submission and archiving of clinical trials data for medical and biopharmaceutical product development. The FDA has collaborated with CDISC to standardize the content and structure of clinical trials and non-clinical study data for regulatory submission. CDISC sponsors and members represent more than 150 companies active in the research and development of regulated health-related products.

1.3.1. Operational Data Model (ODM)

The define.xml standard is based on the CDISC Operational Data Model (ODM). The ODM is a specification of a standard XML schema for the interchange and archive of clinical trials data and metadata. The ODM was developed by a team representing a wide spectrum of stakeholders in the area of clinical research information systems including representatives from pharmaceutical companies, contract research organizations, software vendors and the FDA. ODM is already in use within the biopharmaceutical industry for interchange of clinical trials data and metadata. By design, the ODM is vendor neutral and platform independent and is compatible with a wide range of current, legacy and emergent clinical trials systems. The current version of the ODM standard is available at http://www.cdisc.org/standards/index.html.

To address the specific needs of data transmission in support of regulatory submissions, some extensions have been made to the base ODM schema. These extensions follow the guidelines for Vendor Extensions provided in the ODM specification. The define.xml schema is available online at http://www.cdisc.org/schema/define1-0-0.xsd and has been included within the appendix section of this document.

The combination of ODM and data definition schema structures used in the define.xml are intended to meet or exceed the minimum requirements for electronic transmission of metadata of a wide range of standards including, but not limited to the CDISC Study Data Tabulation Model (SDTM), CDISC Analysis Dataset Model (ADaM), and Standard for Exchange of Non-clinical Data (SEND). While this document is intended to be understandable to readers with minimal technical knowledge of the ODM or of XML, interested readers may find it helpful to refer to the ODM specification provided with the ODM download package available online at http://www.cdisc.org/standards/index.html.

1.3.2. Study Data Tabulation Model (SDTM)

The CDISC Study Data Tabulation Model (SDTM) defines a standard structure for data tabulations that are to be submitted as part of a product application to a regulatory authority such as the FDA. The CDISC SDTM will be used to submit clinical trial and non-clinical study data for product applications across all therapeutic areas. To accommodate this broad range of use, the model provides flexibility in two significant ways. First, each data class defines a superset of variables (an exclusive list) that could be used to define a specific domain where only a small set of core items are required to be included in every submission. Second, the set of domains required to be included within a specific application is not prescribed by the standard. It is determined by the details of a trial protocol and negotiated by the application sponsor working directly with the applicable regulatory agency.

The CDISC SDTM domain and variable metadata should be provided via the CRT DD Specification. See http://www.cdisc.org/standards/index.html for the latest version of the CDISC SDTM standard.

1.3.3. Analysis Dataset Model (ADaM)

The CDISC Analysis Dataset Model (ADaM) defines a standard for analysis datasets that are to be submitted in addition to the data tabulation (CDISC SDTM) datasets as described above in Section 1.3.2. These models provide clear and unambiguous communication of the content,

source, and quality of the datasets submitted in support of the statistical analyses performed by the sponsor. In most implementations, the analysis datasets will contain variables from multiple CDISC SDTM datasets and derived variables for specific analyzes. These datasets should be useable by currently available tools, while providing machine-readable metadata to facilitate future standard analysis tool development. In 2004, ADaM posted the General Considerations for Analysis Datasets, followed by additional models for specific statistical analyses. See http://www.cdisc.org/standards/index.html for the current ADaM models. Like the CDISC SDTM, the analysis domain and variable metadata should also be provided via the CRT DD Specification.

2. The Case Report Tabulation Data Definition Specification

2.1. The Structure of the Data Definition Document

The Data Definition document (define.pdf) has two main parts, a Table of Contents (TOC) and a collection of Data Definition Tables. The TOC lists all of the datasets (also known as domains) included in the submission per study or collection of studies (e.g., in support of the Summary of Clinical Safety or Summary of Clinical Efficacy). Whereas the Data Definition Tables provide detailed information about the variables contained within each of the domains.

In a define.xml file, an ODM element known as the ItemGroupDef is used to provide TOC domain level metadata, whereas the ItemDef is used to define the main metadata of the variables included within the domains.

2.1.1. The Table of Contents (TOC)

Every dataset contained within a submission (e.g., Tabulation, Analysis), should be described using the CDISC Dataset Metadata fields listed below.

2.1.1.1. Domain Level Metadata

Field	Description
Dataset	The file name of the dataset or data domain name (e.g., "DM.xpt", "DM")
Description	A short description of the type of information contained within the dataset (e.g. "Demographics", "Quality of Life Analysis Data").
Structure	The level of detail represented by individual records in the dataset (e.g., "One record per subject", "One record per subject per visit", "One record per subject per event").
Purpose	Purpose for the dataset (e.g., "Tabulation", "Analysis").
Keys	Used to uniquely identify and index each record in a dataset and could function as foreign keys to facilitate linking to other datasets. Most datasets will have between 2 and 5 key variables.
Location	Folder and filename where the dataset can be found.

2.1.1.2. An example Table of Contents

Datasets for Study 1234					
Dataset	Description	Structure	Purpose	Keys	Location
TE	Trial Elements	One record per element	Tabulation	STUDYID, DOMAIN, ETCD	crt/datasets/1234/te.xpt
AE	Adverse Events	One record per subject per event	Tabulation	USUBJID, AETERM, AESEQ	crt/datasets/1234/ae.xpt
VSST	Vital Signs Statistical Analysis	One record per subject per visit	Analysis	USUBJID, VISIT	crt/datasets/1234/vsst.xpt

2.1.1.3. CDISC Dataset Metadata in XML

The domain level metadata is to be represented by an ODM ItemGroupDef element using the attributes and sub-elements as shown below:

XML Field*	Submission Metadata Field	Status**	Description
ItemGroupDef Attributes:			
OID	N/A	Required	The unique ID of the domain and could be the same as Name.
Name	Dataset	Required	The file name of the dataset or data domain name (e.g., "DM" for Demography data).
Repeating	N/A	Required	Valid values are "Yes" or "No". "Yes" is for domains with more than one record per subject whereas, "No" is for domains with 1 record per subject.
IsReferenceData	N/A	Optional	Valid values are "Yes" or "No". "Yes" is for datasets that contain reference data only, no subject level data (e.g., Trial Design domains or Laboratory Reference Ranges). "No" indicates subject level data. Absence of this attribute indicates subject level data.
Purpose	Purpose	Optional	Purpose of the data (e.g., "Tabulation", "Analysis").
def:Label	Description	Required	Brief description of the data domain.
def:Structure	Structure	Optional	Data domain structure.
def:DomainKeys	Keys	Optional	Comma separated text string that contains the dataset key variables.
def:Class	N/A	Optional	General class of the data domain (e.g., "Events", "Interventions", "Findings", "Special Purpose", "Analysis", "Selection", "Support", and "Legacy").
def:ArchiveLocationID	N/A	Required	Reference to the def:leaf that links to the location of the dataset.

XML Field*	Submission Metadata Field	Status**	Description
ItemGroupDef Child Elements:			
ItemRef	N/A	Required	Variable reference of the item definition. See Section 2.1.2.3 for the ItemRef Attributes.
def:leaf	N/A	Required	Contains the XLink information that def:ArchiveLocationID references.
def:leaf Attributes:			
ID		Required	Leaf identifier. Unique ID that is referenced by def:ArchiveLocationID.
xlink:href	N/A	Required	Link to the domain file. Should be the pathname and filename of the target dataset relative to the define.xml.
def:leaf Child Elements:			
def:title	Location	Required	Meaningful description, label, or location of the domain leaf (e.g., "crt/datasets/Protocol 1234/AE.xpt").

* The element ordering as described above should be adhered to within the define.xml.
** Required elements and attributes are always to be included in the define.xml.

2.1.1.4. Example of ItemGroupDef

```
<ItemGroupDef OID="docroot.IG.TE"
     Name="TE"
     Repeating="Yes"
     IsReferenceData="Yes"
     Purpose="Tabulation"
     def:Label="Trial Elements"
     def:Structure="One record per element"
     def:DomainKeys="STUDYID, DOMAIN, ETCD"
     def:Class="Trial Design"
     def:ArchiveLocationID="ArchiveLocation.te">
    <!-- All ItemRefs would be listed here -->
    <def:leaf ID="ArchiveLocation.te"
        xlink:href="te.xpt">
        <def:title>te.xpt</def:title>
    </def:leaf>
</ItemGroupDef>
<ItemGroupDef OID="docroot.IG.AE"
     Name="AE"
     Repeating="Yes"
     IsReferenceData="No"
     Purpose="Tabulation"
     def:Label="Adverse Events"
     def:Structure="One record per subject per event"
     def:DomainKeys="USUBJID, AETERM, AESEQ"
     def:Class="Events"
     def:ArchiveLocationID="ArchiveLocation.ae">
    <!-- All ItemRefs would be listed here -->
    <def:leaf ID="ArchiveLocation.ae"
        xlink:href="ae.xpt">
        <def:title>ae.xpt</def:title>
    </def:leaf>
</ItemGroupDef>
<ItemGroupDef OID="docroot.IG.VSST"
     Name="VSST"
     Repeating="Yes"
     IsReferenceData="No"
     Purpose="Analysis"
     def:Label="Vital Signs Statistical Analysis"
     def:Structure="One record per subject per visit"
     def:DomainKeys="USUBJID, VISIT"
     def:Class="Analysis"
     def:ArchiveLocationID="ArchiveLocation.vsst">
    <!-- All ItemRefs would be listed here -->
    <def:leaf ID="ArchiveLocation.vsst"
        xlink:href="vsst.xpt">
        <def:title>vsst.xpt</def:title>
    </def:leaf>
</ItemGroupDef>
```

def:ArchiveLocationID must match the def:leaf/@ID

def:ArchiveLocationID must match the def:leaf/@ID

def:ArchiveLocationID must match the def:leaf/@ID

2.1.1.5. An Explanation of the ItemGroupDef Examples

The value of ItemGroupDef/@OID and def:leaf/@ID must be unique within the define.xml file. Furthermore, the ItemGroupDef must contain an ItemRef element for each variable included in the corresponding dataset. The format of the OID and ID attributes is flexible such that sponsors can use their own naming conventions. For example, the above ItemGroupDef/@OID (example 2.1.1.4) and the ItemRef/@ItemOID in examples 2.1.2.6 and 2.1.2.7 concatenated the source dataset name and the variable name (e.g., "docroot.IG.TE", "AE.AESER", and "VS.BMI", respectively).

2.1.2. The Data Definition Tables

The Data Definition Table section provides the variable level attributes, definitions, and usage information for each variable contained within each domain. This section should be described using the CDISC Variable Metadata fields listed below:

2.1.2.1. CDISC Variable Metadata Fields

Field	Description
Variable	The field name of the variable.
Label	A brief description of the variable.
Type	The variable type (e.g., "text", "float", "date"); see Section 2.1.2.4 for more details.
Controlled Terms or Format	The set of controlled terms or variable display information.
Origin	Indicator of the origin of the variable. Examples could include Case Report Form (CRF) Page numbers, derived, or a reference to other variable(s) (e.g., DM.AGE).
Role	Information on how a variable is used within the dataset (e.g., "Identifier", "Topic", "Timing", "Qualifier", "Selection", "Analysis").
Comment	Other information regarding the variable definition, usage, etc.

2.1.2.2. An example Data Definition Table

Dataset Variables for Adverse Events (AE)						
Variable	**Label**	**Type**	**Controlled Terms or Format**	**Origin**	**Role**	**Comment**
STUDYID	Study Identifier	text		CRF Page	Identifier	
DOMAIN	Domain Abbreviation	text	AE	CRF Page	Identifier	
USUBJID	Unique Subject Identifier	text		CRF Page	Identifier	
AESEQ	Sequence Number	integer		Derived	Identifier	
AETERM	Reported Term for the Adverse Event	text		CRF Page	Topic	
AEDECOD	Dictionary-Derived Term	text		Derived	Synonym Qualifier	

2.1.2.3. CDISC Dataset Contents in XML

The metadata provided in the Data Definition Table should be represented using the ODM ItemRef and ItemDef.

2.1.2.3.1. ItemRef

XML Field*	Submission Metadata Field	Status**	Additional Description
ItemRef Attributes			Referred from Section 2.1.1.3.
ItemOID	N/A	Required	Reference to the unique ID of the variable.
OrderNumber	N/A	Optional	Provides an ordering to the ItemRefs. Values of OrderNumber must be unique within each Domain's ItemGroupDef.
Mandatory	N/A	Required	Valid values are "Yes" and "No". "Yes", indicates that the clinical data for an instance of the containing item group is required or would be incomplete without an instance of this item.
Role	Role	Optional	Space delimited variable classification (e.g., "Identifier", "Topic", "Qualifier", "Timing", "Selection", "Analysis"). Values are interpreted as codes from the CodeList specified by @RoleCodeListOID. Variables may have multiple roles.
RoleCodeListOID	N/A	Optional	The OID of the corresponding Role Code List. Provides decodes for the codes given in @Role. This attribute should be present if and only if the Role attribute is.

* The element ordering as described above should be adhered to within the define.xml.
** Required elements and attributes are always to be included in the define.xml.

2.1.2.3.2. ItemDef

XML Field*	Submission Metadata Field	Status**	Additional Description
ItemDef Attributes			
OID	N/A	Required	The unique OID of the Variable.
Name	Variable	Required	Variable name.
DataType	Type	Required	Valid values are "text", "integer", "float", "date", "time" and "datetime". See Section 2.1.2.4 for more information.
Length	N/A	Conditional	The variable length. Required if Type is "text", "integer", or "float".
SignificantDigits***	N/A	Conditional	The number of decimal digits. Required if Type is "float".
Origin	Origin	Optional	Values such as "CRF Page numbers", "derived", or variable references.
Comment	Comment	Required	Further descriptions and/or information regarding the variable.
def:Label	Label	Required	Variable label.
def:DisplayFormat	N/A	Optional	Display format for numeric variables (e.g., 8.2 means display floating point variable values to the second decimal place). Can be displayed in the "Controlled Terms or Format" column.
def:ComputationMethod OID	N/A	Optional	The OID of the corresponding ComputationMethod.

* The element ordering as described above should be adhered to within the define.xml.
** Required elements and attributes are always to be included in the define.xml.
*** The meaning of the ODM attribute SignificantDigits, as of this publish, should be considered as DecimalDigits. Future versions of ODM and/or define.xml may address this nomenclature directly, and the right/intent to do so is thusly reserved.

XML Field*	Submission Metadata Field	Status**	Additional Description
ItemDef Child Elements			
CodeListRef	N/A	Optional	CodeList associated with the variable. Required if the variable has a code-decode association or optional if it has a discrete set of values.
def:ValueListRef		Optional	ValueList associated with the variable; see Section 2.1.4 for more information.
CodeListRef Attribute			
CodeListOID	N/A	Conditional	The OID of the corresponding CodeList.
def:ValueListRef Attribute			
def:ValueListOID		Conditional	The OID of the corresponding def:ValueListDef; see Section 2.1.4 for more information.

* The element ordering as described above should be adhered to within the define.xml.
** Required elements and attributes are always to be included in the define.xml.

2.1.2.4. Data Type Considerations

The CDISC SDTM and some software applications classify variables into one of two types, character and numeric ("Char" and "Num", respectively), however ODM supports 6 data types ("text", "integer", "float", "date", "time" and "datetime"). The following recommendations are to be used for representing variable information using the ODM DataType and Length attributes:

ODM Type**	Other Type*	Length	SignificantDigits	Considerations
text	Char	Actual maximum length of variable or a maximum value of 32768.	N/A	If the data field were 200 characters (e.g., the SAS Version 5 transport file variable length restriction) then the length would be 200.
integer	Num	The largest integer width.	N/A	Used for numeric or equivalent variables that have discrete whole values (non-fractional) they can be positive, negative, or zero.
float	Num	The largest whole number width plus the maximum number of decimal digits.	The number of decimal digits	Use for other non-integer variables where the data type is numeric or equivalent; see the ODM documentation for the significant digit representation.
datetime	Char	N/A	N/A	Use if values for variable represent complete Date Times (YYYY-MM-DDT HH:MM:SS.SS).
date	Char	N/A	N/A	Use if values for variable represent complete (YYYY-MM-DD) dates.
time	Char	N/A	N/A	Use if values for variable represent complete (HH:MM:SS.SS) times.

* The "Other Types" are examples and do not necessarily represent an exclusive list of data types that could be mapped to the ODM Data Types.

** The above formats for *dates*, *times*, and *datetimes* are compatible with ISO 8601.

Note: HL7 represents integer and floating-point variables as “string literals” or true floating-point numbers (base 2, 8, 16, or 64) and the relationship between the variable length and precision depends on the representation chosen.

2.1.2.5. First Variable Example

```
<ItemGroupDef OID="DM" …>
   <ItemRef ItemOID="USUBJID"
      OrderNumber="1"
      Mandatory="Yes"
      Role="IDENTIFIER"
      RoleCodeListOID="RoleCodeList"/>
</ItemGroupDef>
<ItemDef OID="USUBJID"
   Name="USUBJID"
   DataType="text"
   Length="80"
   Origin="CRF Page"
   Comment="Demographics CRF Page 4"
   def:Label="Unique Subject Identifier">
</ItemDef>
```

ItemOID must match the ItemDef/@OID

2.1.2.6. Second Variable Example

```
<ItemGroupDef OID="AE" …>
   <ItemRef ItemOID="AE.AESER"
      OrderNumber="10"
      Mandatory="Yes"
      Role="QUALIFIER"
      RoleCodeListOID="RoleCodeList"/>
</ItemGroupDef>

<ItemDef OID="AE.AESER"
   Name="AESER"
   DataType="text"
   Length="8"
   Origin="CRF Page"
   Comment="Adverse Event Seriousness"
   def:Label="Seriousness of AE Code">
   <CodeListRef CodeListOID="CodeList.AESER"/>
</ItemDef>


<CodeList OID="CodeList.AESER"
   Name="AESER"
   DataType="text"
   SASFormatName="$AESER">
   <CodeListItem CodedValue="N">
      <Decode>
         <TranslatedText xml:lang="en">No</TranslatedText>
      </Decode>
   </CodeListItem>
   <CodeListItem CodedValue="Y">
      <Decode>
         <TranslatedText xml:lang="en">Yes</TranslatedText>
      </Decode>
   </CodeListItem>
</CodeList>
```

ItemOID must match the ItemDef/@OID

CodeListOID must match the CodeList/@OID

2.1.2.7. Third Variable Example

```
<ItemGroupDef OID="VS" …>
<ItemRef ItemOID="VS.BMI"
   OrderNumber="20"
   Mandatory="Yes"
   Role="ANALYSIS"
   RoleCodeListOID="RoleCodeList"/>
</ItemGroupDef>

<ItemDef OID="VS.BMI"
   Name="BMI"
   DataType="float"
   Length="6
   SignificantDigitis="3"
   Origin="Derived"
   Comment="Calculated"
   def:Label="Body Mass Index"
   def:DisplayFormat="8.3"
   def:ComputationMethodOID="BMI">
</ItemDef>

<def:ComputationMethod OID="BMI">WEIGHT DIVIDED BY HEIGHT TIMES HEIGHT
</def:ComputationMethod>
```

ItemOID must match the ItemDef/@OID

def:ComputationMethodOID must match the def:ComputationMethod/@OID

2.1.2.8. Fourth Variable Example

```
   <ItemGroupDef OID="DM" …>
      <ItemRef ItemOID="USUBJID" OrderNumber="1" Mandatory="Yes" Role="IDENTIFIER"
        RoleCodeListOID="RoleCodeList"/>
       <!-- Additional variables would be listed here -->
    </ItemGroupDef>
   <ItemGroupDef OID="EX" …>
      <ItemRef ItemOID="USUBJID" OrderNumber="1" Mandatory="Yes" Role="IDENTIFIER"
        RoleCodeListOID="RoleCodeList"/>
       <!-- Additional variables would be listed here -->
    </ItemGroupDef>
   <ItemGroupDef OID="VS" …>
      <ItemRef ItemOID="USUBJID" OrderNumber="1" Mandatory="Yes" Role="IDENTIFIER"
        RoleCodeListOID="RoleCodeList"/>
       <!-- Additional variables would be listed here -->
    </ItemGroupDef>

<ItemDef OID="USUBJID"
   Name="USUBJID"
   DataType="text"
   Length="20"
   Origin="CRF Page"
   Comment="Unique subject identifier within the submission."
   def:Label="Unique Subject Identifier">
</ItemDef>
```

Each ItemOID must match the corresponding ItemDef/@OID

2.1.2.9. An Explanation of the ItemDef Examples

The above examples contain the necessary information to represent the respective variables in the Data Definition Table section of the define document. The first example (Section 2.1.2.5) illustrates how an ItemDef/@OID contains the variable attribute and definition information of a simple variable and how it is referenced by an ItemRef/@ItemOID. It should be noted that each ItemDef/@OID must be unique and the ItemOID attribute in ItemRef must match the OID in the corresponding ItemDef. The second example (Section 2.1.2.6) defines a variable whose values are limited to a set of controlled terms. It contains a reference to a code list (CodeListRef/@CodeListOID) and shows the definition of the corresponding CodeList element. The third example (Section 2.1.2.7) illustrates a variable whose values are derived according to an algorithm defined in a def:ComputationMethod element (in this case for computation of Body Mass Index). The computation method can be a mathematical formula or a textual description of the algorithm used to perform the computation. If the a variable is used in multiple datasets and has identical attributes (on the ItemDef), then there should only be a single ItemDef, which is referenced by an ItemRef in each dataset it is contained in. This is exemplified in the fourth and final example (Section 2.1.2.8), where the variable USUBJID is contained within three separate datasets (DM, EX, and VS). It has the same definition in each dataset, and therefore each ItemRef/@ItemOID value references the same ItemDef/@OID. The attributes of each of those ItemRefs can however be different, for example the same variable may be deemed mandatory in one dataset but not in another.

2.1.3. Controlled Terminology (Code Lists)

Many variables have a discrete list of valid values or controlled terms associated with them. Controlled terminologies may be developed by sponsors or by an external third party such as MedDRA. The ODM CodeList element is used to define controlled terminologies contained within a submission. For variables whose values are limited to a discrete set of valid values, the corresponding ItemDef should include a CodeListRef element that references a CodeList element included within the define.xml file.

The ODM CodeList element may define either an internal or external CodeList. Internal CodeLists include a list of allowable codes and their corresponding decodes. For coded values that represent an order that is meaningful in an analysis or display context, the rank order should be provided. External CodeLists corresponding to controlled terminologies provided by external third parties are identified using an ExternalCodeList element that identifies the dictionary by name and version.

A simple example of a code list is 'YesNo' that contains two code/decode pairs: Y/Yes, N/No, respectively. Another example of a CodeList is where the codes may have an ordinal or rank value as seen in the Adverse Events severity CodeList table below (see example 2.1.3.3 for the corresponding XML representation):

CodedValue	def:Rank	Decode
MILD	1.0	Mild
MOD	2.0	Moderate
SEV	3.0	Severe

2.1.3.1. Controlled Terminologies Represented in XML

Code list/formats maps to the XML CodeListItem element information as shown below:

XML Field*	Status**	Description
CodeList Attributes:		
OID	Required	The unique ID of the Code List.
Name	Required	Short name of the Code List.
DataType	Required	The Type of the codes. Valid values are "text", "float", and "integer".
SASFormatName	Optional	SAS format name.
CodeList Child Elements		
CodeListItem	Conditional	An entry in the CodeList. Either a CodeList or ExternalCodeList element is required.
ExternalCodeList	Conditional	Details of an external CodeList Either a CodeList or ExternalCodeList element is required.
CodeListItem Attributes		
CodedValue	Required	The coded value.
def:Rank	Optional	Rank order of decodes.
CodeListItem Child Elements		
Decode	Required	The decode value where all CodeListItems must have a decode.
Decode Child Elements		
TranslatedText	Required	The 'text' translation of the code.
TranslatedText Attributes		
xml:lang	Optional	Language of the decode text.
ExternalCodeList Attributes		
Dictionary	Optional	Name of the dictionary.
Version	Optional	Dictionary version.

* The element ordering as described above should be adhered to within the define.xml.

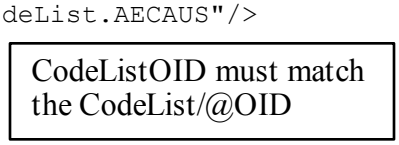
** Required elements and attributes are always to be included in the define.xml.

2.1.3.2. First CodeList Example (Internally Controlled Terms)

An example of a simple numeric code to text decode format:

```
<ItemDef OID="VAR.AECAUS"
   Name="AECAUS"
   DataType="text"
   Length="200"
   Origin="CRF Page"
   Comment="None"
   def:Label="Adverse Event Causality">
   <CodeListRef CodeListOID="CodeList.AECAUS"/>
</ItemDef>

<CodeList OID="CodeList.AECAUS"
   Name="AECAUS"
   DataType="integer">
   <CodeListItem CodedValue="1">
      <Decode>
         <TranslatedText xml:lang="en">DISEASE UNDER STUDY</TranslatedText>
      </Decode>
   </CodeListItem>
   <CodeListItem CodedValue="2">
      <Decode>
         <TranslatedText xml:lang="en">CONCOMITANT TRT</TranslatedText>
      </Decode>
   </CodeListItem>
</CodeList>
```



2.1.3.3. Second CodeList Example (With Ranking)

In some cases, the coded values may have an implied ordering or ranking that is meaningful in a data analysis or display context. For example, the severity of an Adverse Event may be described using a CodeList with values 'MILD', 'MOD' 'SEV' which are ranked 1, 2, and 3, respectively.

```
<ItemDef OID="VAR.AESEVTXT"
   Name="AESEVTXT"
   DataType="text"
   Length="200"
   Origin="CRF Page"
   Comment="None"
   def:Label="Severity/Intensity">
   <CodeListRef CodeListOID="CodeList.AESEVTXT"/>
</ItemDef>
```


```
<CodeList OID="CodeList.AESEVTXT"
   Name="AECAUS"
   DataType="text"
   SASFormatName="$AESEV">
   <CodeListItem CodedValue="MILD" def:Rank="1.0">
      <Decode>
         <TranslatedText xml:lang="en">Mild</TranslatedText>
      </Decode>
   </CodeListItem>
   <CodeListItem CodedValue="MOD" def:Rank="2.0">
      <Decode>
         <TranslatedText xml:lang="en">Moderate</TranslatedText>
      </Decode>
   </CodeListItem>
   <CodeListItem CodedValue="SEV" def:Rank="3.0">
      <Decode>
      <TranslatedText xml:lang="en">Severe</TranslatedText>
      </Decode>
   </CodeListItem>
</CodeList>
```

2.1.3.4. Third CodeList Example (A Discrete List of Values)

The CodeList element could be used to define those variables that have a discrete list of valid values. An example CodeList:

```
<ItemDef OID="VAR.TESTCODE"
   Name="TESTCODE"
   DataType="text"
   Length="200"
   Origin="CRF Page"
   Comment="None"
   def:Label="A TESTCODE Example">
   <CodeListRef CodeListOID="DiscreteList.TESTCODE"/>
</ItemDef>
```

CodeListOID must match the CodeList/@OID

```
<CodeList OID="DiscreteList.TESTCODE"
   Name="TESTCODE"
   DataType="text">
   <CodeListItem CodedValue="Value 1">
      <Decode>
         <TranslatedText>Value 1</TranslatedText>
      </Decode>
   </CodeListItem>
   <CodeListItem CodedValue="Value 2">
      <Decode>
         <TranslatedText>Value 2</TranslatedText>
      </Decode>
   </CodeListItem>
</CodeList>
```

2.1.3.5. Fourth CodeList Example (External CodeList)

For variables whose values come from a controlled vocabulary provided by a third party the CodeListRef refers to an ExternalCodeList. The example below shows a CodeList element referring to version 6 of MedDRA.

```
<ItemDef OID="VAR.AEBODSYS"
   Name="AEBODSYS"
   DataType="text"
   Length="200"
   Origin="CRF Page"
   Comment="None"
   def:Label="Body System or Organ Class">
  <CodeListRef CodeListOID="ExternalCodeList.AEBODSYS"/>
</ItemDef>
```

CodeListOID must match the CodeList/@OID

```
<CodeList OID="ExternalCodeList.AEBODSYS"
   Name="MedDRA Adverse Events Dictionary"
   DataType="text">
   <ExternalCodeList Dictionary="MedDRA" Version="6.0"/>
</CodeList>
```

2.1.3.6. An Explanation of the CodeList Examples

In the first example, the variable AECAUS has been coded as 1 and 2 (this corresponds to CodedValue) and the textual value (or replacement value – decode) is contained in TranslatedText of "DISEASE UNDER STUDY" and "CONCOMITANT TRT", respectively. The second example is similar to the first in that it describes the code-decode relationship, however, the code values are text and these textual values have an implied ordinal rank value defined by the CodeListItem Rank attribute. The third example illustrates a CodeList that serves to define a set of valid values whereas the last example (4th) illustrates how to reference an external code list or dictionary.

2.1.4. Value Level Metadata (Value List)

Case Report Tabulation datasets, such as those defined by the CDISC SDTM, that have normalized data structure (e.g., one record per patient per test code per visit) provide an efficient method for the interchange of information mainly because the data structure does not change when new information is added. However, this structure requires additional metadata support in order to be useful for review and analysis.

For example, the Vital Signs data domain as defined by the CDISC SDTM Version 3.1 may contain subject records related to height, weight, and frame size, which would be stored within a structure containing one row per subject per vital signs measurement per visit. Each subject therefore could have three records per visit. Per the CDISC SDTM, the measurement parameter name is stored in the test code variable (VSTESTCD) and the parameter value is stored in a result variable (e.g., VSORRES, VSSTRESN or VSSTRESC) as shown in the table below:

USUBJID	VSTESTCD	VSTEST	VSORRES	VSSTRESN	VSSTRESC	VISITNUM
00001	HEIGHT	Height in cm	183	183		1
00001	WEIGHT	Weight in kg	88.5	88.5		1
00001	FRAME	Body Frame Size	Large		Large	1

As shown in this table, the values of the results corresponding to different test codes have different attributes (e.g., data types, definitions, or display formats). To communicate the definitions and attributes of these variable values clearly, value level metadata should be provided for each unique value of the test code variable (e.g., VSTESTCD). This value level metadata could also be used for transforming the data into other structures to enhance review and analysis – that is to a less normalized 'horizontal' data structure. This metadata should be provided as an appendix section within the Data Definition Document within a table similar in structure previously described in Section 2.1.2.2 and is illustrated below.

An example Value Level Metadata within the define.pdf:

Value Level Metadata								
Source Dataset	Source Variable	Value	Label	Type	Controlled Terms or Format	Origin	Role	Comment
VS	VSTESTCD	HEIGHT	Height in cm	float	3.0	CRF Page 5		
VS	VSTESTCD	WEIGHT	Weight in kg	float	5.1	CRF Page 5		
VS	VSTESTCD	FRAME	Frame	text	Small, Medium, Large	CRF Page 6		

Alternatively, the Value Level Metadata could be presented separately by each contributing dataset instead of placing all information in one table. In this case, place the dataset description and/or name within the title (e.g., "Value Level Metadata for Subject Characteristics (SC)").

2.1.4.1. Value List Represented in XML

The def:ValueListRef/@def:ValueListOID attribute is contained within the ItemDef (see Section 2.1.2.3 for explanation). For each unique Test Code the corresponding def:ValueListDef contains an ItemRef element. The ItemRef element includes an ItemDef/@OID attribute that references an ItemDef containing the metadata for that test.

XML Field*	Status**	Description
def:ValueListDef Attributes:		
OID	Required	The unique ID of the Value List.
def:ValueListDef Child Elements:		
ItemRef	Required	The ID of the corresponding Item.

* The element ordering as described above should be adhered to within the define.xml.
** Required elements and attributes are always to be included in the define.xml.

2.1.4.2. A Value Level Metadata Example

The value list reference is contained within the ItemDef as illustrated below:


```
<def:ValueListDef OID="VS01">
   <ItemRef ItemOID="HEIGHT" OrderNumber="1" Mandatory="No"/>
   <ItemRef ItemOID="WEIGHT" OrderNumber="2" Mandatory="No"/>
   <ItemRef ItemOID="FRAME" OrderNumber="4" Mandatory="No"/>
</def:ValueListDef>

<ItemDef OID="VSTESTCD" Name="VSTESTCD" DataType="text"
   Length="200" Origin="CRF Page" Comment="None"
   def:Label="Vital Signs Test Name">
   <def:ValueListRef def:ValueListOID="VS01"/>
</ItemDef>

<ItemDef OID="HEIGHT" Name="Height" DataType="integer" Length="8"
   Origin="CRF Page 5" def:Label="Height in cm"
   def:DisplayFormat="8.0">
</ItemDef>
<ItemDef OID="WEIGHT" Name="Weight" DataType="float" Length="8"
   SignificantDigits="2" Origin="CRF Page 5"
   def:Label="Weight in kg" def:DisplayFormat="8.1">
</ItemDef>
<ItemDef OID="FRAME" Name="Frame" DataType="text" Length="12"
   Origin="CRF Page 6" def:Label="Body Frame Size">
</ItemDef>
```

2.1.4.3. An Explanation of the Value Level Metadata Example

The example represents the implementation of the previous vital signs example where VSTESTCD has three discrete values (HEIGHT, WEIGHT, and FRAME). This is represented in the ItemDef for VSTESTCD by a def:ValueListRef attribute that refers to a def:ValueListDef (Arrow A) element. The def:ValueListDef contains ItemRefs, one for each unique Test Code. Each of the ItemRefs is linked to a corresponding ItemDef, which defines the metadata for that specific Test Code (Arrow B). It should be noted that due to the XML extension mechanism the def:ValueListDef must precede the ItemDefs.

2.1.5. Computational Algorithms

The def:ComputationMethod element is to be used to provide details about computational algorithms used to derive or impute variable values. The most common use of def:ComputationMethod is to explain how values were generated for derived variables using an algorithm. The algorithm should define the source variable(s) and the calculations performed and usually involves all or a substantial subset of observations within a dataset. The second use of def:ComputationMethod, primarily used for analysis datasets, is to describe how to impute values in place of missing values. In most cases, a separate "flag" variable is set to indicate which observations within the clinical data are derived or imputed.

2.1.5.1. A Computation Method Example

```
<ItemDef OID="VS.MEANBP"
         Name="MEANBP"
         DataType="float"
         Length="8"
         SsignificantDigits="2"
         Origin="VSSTRESN"
         def:Label="Mean Blood Pressure"
         def:DisplayFormat="8.2"
         def:ComputationMethodOID="COMPMETHOD.MEANBP">
</ItemDef>
```

def:ComputationMethodOID must match the def:ComputationMethod/@OID

```
<def:ComputationMethod OID="COMPMETHOD.MEANBP">Sum all readings (Add VARIABLE
NAME here) for visits 0, 1, and 2 and divide by the total number of readings
</def:ComputationMethod>
```

2.1.5.2. An Explanation of the Computation Method Example

Separating the algorithm details from the individual variable definitions (such as placing them in the comment attribute of ItemDef) enables multiple variables to be able to reference the same definition. In order for an ItemDef to reference the corresponding computation method their must be a match between def:ComputationMethodOID and def:ComputationMethod/@OID.

2.2. ODM XML Header, Study, and MetaDataVersion Sections

2.2.1. define.xml Header Section

The first few lines of the define.xml file are important for XML processing. The first line (as seen in line A in Section 2.2.1.1) identifies the file as an XML document and references the XML specification version. The second line (Section B, in Section 2.2.1.1) defines the ODM root element. The ODM root element includes ODM processing information, human readable identification information and identifies the namespaces referenced in the document. The ODM root attributes are:

XML Field*	Status**	Description
ODM Attributes:		
FileType	Required	Type of transmission. Valid value is "Snapshot".
FileOID	Required	An unique identifier for this file. See the ODM Specification for a discussion of FileOID recommendations.
CreationDateTime	Required	Date and Time when the define.xml file was last modified (datetime).
PriorFileOID	Optional	Reference to the previous file (if any).
ODMVersion	Required	The ODM Version number.
Originator	Optional	The organization that generated the define.xml file.
SourceSystem	Optional	The computer system, database management system, etc. that is the source of the define.xml information.
SourceSystemVersion	Optional	The version of the "SourceSystem" above.
ODM Child Element		
Study	Required	A logical collection of domains included within a submission. See Section 2.2.2 for more information,

* The element ordering as described above should be adhered to within the define.xml.
** Required elements and attributes are always to be included in the define.xml.

The default namespace for the define.xml is the URI for the CDISC ODM namespace and the minimal set of namespace references also includes XML Schema Instance (xsi), XLink (xlink) and CDISC Define (def). In the example define.xml, the public copy of the Define schema is referenced in the xsi:schemaLocation attribute. In practice, the schema location reference is optional. Applications that process define.xml files should use the public copies of the schema by default. For additional information about namespaces in XML, see http://www.w3.org/TR/REC-xml-names/.

2.2.1.1. Example define.xml Header

A

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

B

```
<ODM
  xmlns="http://www.cdisc.org/ns/odm/v1.2"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:def="http://www.cdisc.org/ns/def/v1.0"
  xsi:schemaLocation="http://www.cdisc.org/ns/odm/v1.2 define1-0-0.xsd"
  FileOID="Study1234"
  ODMVersion="1.2"
  FileType="Snapshot"
  CreationDateTime="2004-07-28T12:34:13-06:00">
```

2.2.1.2. Viewing the define.xml Document

The define.xml document can be rendered in a format that is human readable if it contains an explicit XML style sheet reference. An initial define.xml style sheet (Version 1.0.0) is available at http://www.cdisc.org/models/def/v1.0/define1-0-0.xsl and is compliant with XSLT Version 1.0 specification (1999). Future releases of the style sheet will expand the available functionality and features (e.g., additional CRF page hypertext linking). Updates will be published and made available through the CDISC website (see: http://www.cdisc.org/standards/index.html).

The style sheet reference should be placed immediately before the ODM root element.

The style sheet reference

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="define1-0-0.xsl"?>
<ODM
  xmlns="http://www.cdisc.org/ns/odm/v1.2"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:def="http://www.cdisc.org/ns/def/v1.0"
  xsi:schemaLocation="http://www.cdisc.org/ns/odm/v1.2 define1-0-0.xsd"
  FileOID="Study1234"
  ODMVersion="1.2"
  FileType="Snapshot"
  CreationDateTime="2004-07-28T12:34:13-06:00">
```

If the define.xml includes the above reference and the corresponding style sheet is available in the same folder as the define.xml file, a browser application will format the output to mirror the data definition document layout as described within this specification. It should be noted that the resulting HTML presentation and the availability and usability of functions will vary depending upon which browser application used. The style sheet release notes provides additional information (see http://www.cdisc.org/models/def/v1.0/define1-0-0.xsl).

2.2.2. CDISC Study Metadata in XML

Study is the first element after the ODM tag and it contains the following global variables.

XML Field*	Status**	Description
Study Attributes:		
OID	Required	The unique ID of the Study.
Study Child Elements:		
GlobalVariables	Required	Contains the global variables.
MetaDataVersion	Required	The MetaDataVersion metadata (see Section 2.2.3).
GlobalVariables Child Elements:		
StudyName	Required	Name of the study.
StudyDescription	Required	Description of the study.
ProtocolName	Required	The protocol Name.

* The element ordering as described above should be adhered to within the define.xml.
** Required elements and attributes are always to be included in the define.xml.

2.2.2.1. An example Study Metadata

```
<Study OID="Study1234">
  <GlobalVariables>
    <StudyName>1234</StudyName>
    <StudyDescription>The description of study 1234</StudyDescription>
    <ProtocolName>Study 1234</ProtocolName>
  </GlobalVariables>
```

2.2.3. MetaDataVersions

The domain and variable definitions included within a submission are, together, known as a MetaDataVersion. The define.xml extends the definition of the ODM MetaDataVersion element to include the data standard and version number that apply to the datasets included. The ODM MetaDataVersion also provides a mechanism for inheriting metadata definitions by referencing a MetaDataVersion defined elsewhere in the define.xml file. An earlier MetaDataVersion is referred to by using the ODM Include element (see the ODM documentation for more information, http://www.cdisc.org/standards/index.html).

2.2.3.1. CDISC MetaDataVersion Metadata in XML

XML Field*	Status**	Description
MetaDataVersion Attributes:		
OID	Required	The unique ID of the MetaDataVersion.
Name	Required	Name of the MetaDataVersion.
Description	Required	The description of the data definition document (e.g. "Study *<insert>*, Data Definitions").
def:DefineVersion	Required	The define.xml schema version used (e.g., "1.0.0").
def:StandardName	Required	Short name of the MetaDataVersion (e.g., "CDISC SDTM", "CDISC SEND", "CDISC ADaM").
def:StandardVersion	Required	The version of an external standard to which the data conforms (e.g., "3.1.0").
MetaDataVersion Child Elements:		
def:AnnotatedCRF	Optional	Contains the DocumentRef for each Annotated CRF. For contents, see Section 2.2.4.
def:SupplementalDoc	Optional	Contains the DocumentRef for each Supplemental Document. For contents, see Section 2.2.5.
def:ComputationMethod	Optional	One per computational algorithm. For contents, see Section 2.1.5.
def:ValueListDef	Optional	Value List Definition that contains the relevant ItemRefs. For contents, see Section 2.1.4.1.
Include	Optional	Link to an earlier MetaDataVersion. The earlier MetaDataVersion may be defined within the same or in a separate ODM file. For contents, see ODM specification.
ItemGroupDef	Required	One per domain included within the study submitted. For contents, see Section 2.1.1.3.
ItemDef	Required	One per unique variable. For contents see, Section 2.1.2.3.2.
CodeList	Optional	One per controlled terminology. For contents, see Section 2.1.3.1.
def:leaf	Optional	Present if either the def:AnnotatedCRFID and/or the def:SupplementalDocID are provided. Contains the XLink information.

* The element ordering as described above should be adhered to within the define.xml.
** Required elements and attributes are always to be included in the define.xml.

XML Field*	Status**	Description
def:leaf Attributes:		
ID	Required	Leaf identifier. Unique ID that is referenced by DocumentRef/@leafID.
xlink:href	Required	Link to the file. Should be the pathname and filename of the target dataset relative to the define.xml.
def:leaf Child Elements:		
def:title	Required	Meaningful description, label, or location of the leaf.
def:AnnotatedCRF Child Element:		
def:DocumentRef	Optional	Contains the leafID of the corresponding referenced Annotated CRF document.
def:SupplementalDoc Child Element:		
def:DocumentRef	Optional	Contains the leafID of the referenced Supplemental Document.
DocumentRef Attribute:		
leafID	Conditional	The unique ID of the referenced Annotated CRF and/or Supplemental Document that will be referenced by a corresponding def:leaf.

* The element ordering as described above should be adhered to within the define.xml.
** Required elements and attributes are always to be included in the define.xml.

2.2.3.2. An Example MetaDataVersion

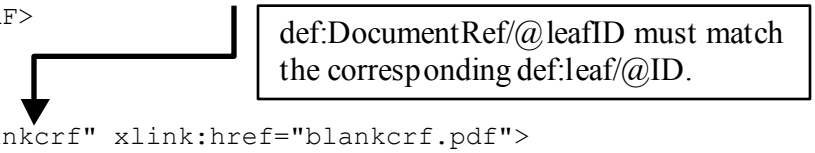
```
<MetaDataVersion OID="CDISC.SDTM.3.1.0"
   Name="CDISC SDTM Data for Study 1234"
   Description="Study 1234, Data Definitions"
   def:DefineVersion="1.0.0"
   def:StandardName="CDISC SDTM"
   def:StandardVersion="3.1.0">
```

2.2.4. Annotated Case Report Form

An Annotated Case Report Form (CRF) is a Portable File Format (PDF) document that maps the data collection fields used to collect subject data to the corresponding variables or discrete variable values contained within the datasets. Within the define.xml, the Annotated CRF is referenced within the def:AnnotatedCRF element of MetaDataVersion. If the def:AnnotatedCRF/@def:DocumentRef attribute is included, there must also be a def:leaf element whose OID attribute value corresponds the def:AnnotatedCRF/@def:DocumentRef attribute value. The associated def:leaf element describes the location of the actual Annotated CRF PDF file relative to the define.xml.

2.2.4.1. An Example of Annotated CRF

```
<def:AnnotatedCRF>
      <def:DocumentRef leafID="blankcrf"/>
</def:AnnotatedCRF>
```




```
<def:leaf ID="blankcrf" xlink:href="blankcrf.pdf">
      <def:title>Annotated Case Report Form</def:title>
</def:leaf>
```

2.2.4.2. An Explanation of the Annotated CRF

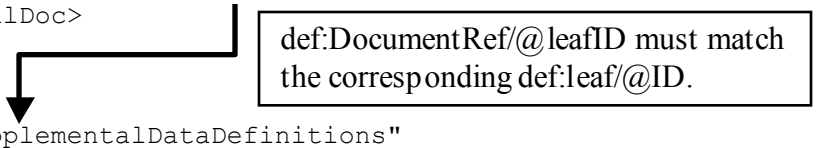
The def:AnnotatedCRF contains one def:DocumentRef/@leafID per Annotated CRF and refers to the def:leaf element, which in turn contains the XLink information of the document.

2.2.5. Supplemental Data Definition Material

Some sponsors provide supplemental data definition information within an appendix section in the Data Definition Document. There are a multitude of reasons why additional or supplemental information is provided chief among them are the desire on the part of the sponsor to provide regulatory reviewers with further explanations or descriptions of the variables contained within the datasets. For example, some clinical algorithms as expressed in the Statistical Analysis Plan (SAP) are complex and may need addition explanations, are best described in a flow chart or in some other graphical depiction. Additionally, the Comment column (within the Data Definition Table) may not be large enough to adequately explain the variable derivation and/or it's usage. In these situations, sponsors often include an appendix section or external PDF file that contains this additional information. Within the define.xml this additional material is referenced within the def:DocumentRef/@leafID of the def:SupplementalDoc element and should be placed within an PDF file (e.g., supplementaldatadefinitions.pdf). The information, however, should not be redundant with any other document contained within the submission (e.g., eCTD Section 16.1.9 contains the "Documentation of statistical methods and interim analysis plans").

2.2.5.1. An Example of Supplemental Data Documentation

```
<def:SupplementalDoc>
      <def:DocumentRef leafID="SupplementalDataDefinitions"/>
</def:SupplementalDoc>
```




```
<def:leaf ID="SupplementalDataDefinitions"
       xlink:href="supplementaldatadefinitions.pdf">
      <def:title>Supplemental Data Definitions</def:title>
</def:leaf>
```

2.2.5.2. An Explanation of The Supplemental Data Documentation

The def:SupplementalDoc contains one def:DocumentRef/@leafID per Supplemental Document that refers to the def:leaf element, which in turn contains the XLink information of the document.

2.3. Conformance to the CRT DD Specification

2.3.1. Well-formed XML

XML is more than placing angle brackets around your data, rather there are specific rules that must be followed when constructing a well-formed XML document. Instead of reiterating the large volume of publicly available XML implementation information in this specification, the following will highlight some of the finer points that are likely to be encountered in the define implementation.

- White space is insignificant (XML Specification §2.10 White Space Handling):
  The define.xml is based upon extensions of the CDISC ODM XML markup specification. More specifically, the ODM is not a text rendering application and white space (blanks, tabs, linefeeds, etc.) is generally not expected to be passed on to a processing application in exactly the same form as entered. Therefore adding spaces for aesthetic reasons is generally irrelevant and ignored by processing applications.

- Special characters must be treated specially (XML Specification §4.1 Character and Entity References and §4.6 Pre-Defined Entities):
  XML has five characters that must be treated specially depending on where they appear in a document; that is, in some places they may not occur in their symbol form since they carry special meaning in XML. The five special characters are:

  ```
  Ampersand          &
  Apostrophe         '
  Quote              "
  Less than          <
  Greater than       >
  ```

  The full rules for where these symbols can and cannot be used can be found in the XML Specification. To keep things simple for the purpose of this guide, the following rule can be used: If any of these characters occur in open text, substitute them as follows:

  ```
  Ampersand          &amp;
  Apostrophe         &apos;
  Quote              &quot;
  Less than          &lt;
  Greater than       &gt;
  ```

  Note: The ampersand character begins each substitution, and the semi-colon ends each substitution string.

  The rendering of other special characters via the XML Character Codepoint Reference mechanism (&#nn; and &#xnn;) is supported within define.xml and the ODM, but best practice is to avoid escapement and use character forms as they occur naturally in the specified XML encoding.

- Namespaces are important:
  Define.xml, as well as any other extension to the ODM, uses a namespace prefix which is normally assigned at the top of the define XML document. This prefix identifies content specific to the individual extension. It is a requirement of the define.xml markup to use the

prefix whenever define-specific content is being added to the document. Best practice assigns "def:" as the prefix of choice for define.xml document content.

2.3.2. Conformance

The define standard is expressed in an XML-Schema representation implementing a CDISC ODM extension. Three files comprise the XML-Schema representation:

- define-ns.xsd
  [Namespace Schema]
  This schema defines the elements and attributes that are included in the define.xml extensions to ODM.
- define-extension.xsd
  [Extension Schema]
  This schema defines where in the ODM the define.xml extensions should appear.
- define1-0-0.xsd
  [Application Extension Schema]
  This schema "glues" the extensions and the ODM together to create the full define.xml definition. Applications implementing define.xml should reference only the define1-0-0.xsd directly.

Conformance in the define.xml context is termed as XML, which is *syntactically, and semantically correct according to the schema expression and the define.xml specification document.*

XML parsers can use the XML-Schema expression to validate that the construction of the file is *syntactically* correct. This is one of the major advantages of the ODM extension mechanism used by the define.xml implementation, as this form removes constructs which made parse time validation of earlier documents complex. However, it is still the responsibility of implementing software systems to validate that the construction of the file is *semantically* correct as not all rules and constraints can be expressed via XML-Schema mechanisms.

The correct ordering of elements within a document is an absolute requirement for the document to be valid with respect to the define.xml schema. The use of an XML-Schema definition and a validating parser environment makes detection of improperly ordered content fairly straightforward. In the absence of such mechanisms, care should be extended to following the order specified by the documentation for all extension content.

There are numerous tools available in the marketplace to assist in validation of XML markup content. For use with the define.xml expression mechanism, the tool must be capable of validating XML-Schema constructs utilizing the *redefine* operator. As of the publication of this specification not all tools support the XML-Schema *redefine* operator, or they may have implementation inconsistencies that prevent them from validating properly content expressed via this mechanism. It is the belief of the define.xml design team that as time progresses these current inconsistencies and limitations will diminish and validating implementations will grow more harmonious.

3. References and Related Documents

1. Dave Christiansen and Wayne Kubick "CDISC Submission Metadata Model", Version 2.0, November 2001 (available at http://www.cdisc.org/pdf/SubmissionMetadataModelV2.pdf)
2. CDISC Operational Data Model, Version 1.2.1, January 2005 (available at http://www.cdisc.org/models/odm/v1.2/index.html).
3. CDISC Study Data Tabulation Model, Version 3.1, June 25, 2004 (available at http://www.cdisc.org/models/sds/v3.1/index.html).
4. CDISC Study Data Tabulation Model Implementation Guide: Human Clinical Trials, Version 3.1, June 25, 2004 (available at http://www.cdisc.org/models/sds/v3.1/index.html).
5. *Code of Federal Regulations,* Title 21, Volume 1, Part 11. Food and Drug Administration. US Government Printing Office, 1998 available at (http://www.fda.gov/cder/guidance/5667fnl.pdf)
6. Food and Drug Administration. *Regulatory Submissions in Electronic Format, General Considerations*. Rockville, MD: US Department of Health and Human Services, Food and Drug Administration; January 1999 (available at http://www.fda.gov/cder/guidance/2867fnl.pdf).
7. Food and Drug Administration Guidance for Industry: Providing Regulatory Submissions in Electronic Format – Human Pharmaceutical Product Applications and Related Submissions; Draft August 2003 (available at http://www.fda.gov/OHRMS/DOCKETS/98fr/5640dft.pdf).
8. Extensible Markup Language (XML) 1.0, Third Edition, 4th February 2004 (available at http://www.w3.org/TR/2004/REC-xml-20040204).

4. CRT Data Definition Schema

- define-ns.xsd: The definition of elements and attributes that have been added via the ODM extension mechanism.
  File:

define-ns.xsd

Contents:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema targetNamespace="http://www.cdisc.org/ns/def/v1.0"
           xmlns="http://www.w3.org/2001/XMLSchema"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
           xmlns:xlink="http://www.w3.org/1999/xlink"
           xmlns:odm="http://www.cdisc.org/ns/odm/v1.2"
           xmlns:def="http://www.cdisc.org/ns/def/v1.0"
           elementFormDefault="qualified" attributeFormDefault="unqualified">
    <xs:import namespace="http://www.w3.org/1999/xlink"
schemaLocation="http://www.oasis-open.org/committees/ebxml-msg/schema/xlink.xsd"/>
    <xs:import namespace="http://www.cdisc.org/ns/odm/v1.2"
schemaLocation="http://www.cdisc.org/schema/odm/v1.2.1/ODM1-2-1.xsd"/>

   <!-- COMMENT: MetaDataVersion
             We are purposefully using xs:id and xs:idref on the def:leaf and
             def:DocumentRef constructs to force parsers to validate whether these
             intended match pairs are actually present in the define file. under
             standard ODM usage, the OID and OIDref are NOT schema ID types to
             circumvent this checking.
     -->

   <xs:complexType name="DEFINEcomplexTypeDefinition-DocumentRef">
     <xs:simpleContent>
       <xs:extension base="string">
            <xs:attribute name="leafID" type="xs:IDREF" use="required"/>
       </xs:extension>
     </xs:simpleContent>
   </xs:complexType>

   <xs:element name="DocumentRef" type="def:DEFINEcomplexTypeDefinition-DocumentRef"/>

       <xs:element name="AnnotatedCRF">
          <xs:complexType>
             <xs:sequence>
                <xs:element ref="def:DocumentRef" minOccurs="1" maxOccurs="unbounded"/>
             </xs:sequence>
          </xs:complexType>
       </xs:element>

       <xs:element name="SupplementalDoc">
          <xs:complexType>
             <xs:sequence>
                <xs:element ref="def:DocumentRef" minOccurs="1" maxOccurs="unbounded"/>
             </xs:sequence>
          </xs:complexType>
       </xs:element>
```

```
      <xs:attribute name="DefineVersion" type="string"/>
              <xs:attribute name="StandardName" type="string"/>
              <xs:attribute name="StandardVersion" type="string"/>

      <xs:element name="leaf">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="title" minOccurs="1" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="ID" type="xs:ID" use="required"/>
            <xs:attribute ref="xlink:href" use="required"/>
         </xs:complexType>
      </xs:element>

       <xs:element name="ComputationMethod">
      <xs:complexType>
        <xs:simpleContent>
          <xs:extension base="string">
             <xs:attribute name="OID" type="odm:oid" use="required"/>
          </xs:extension>
        </xs:simpleContent>
       </xs:complexType>
       </xs:element>

       <xs:element name="ValueListDef">
          <xs:complexType>
             <xs:sequence>
                <xs:element ref="odm:ItemRef" minOccurs="1" maxOccurs="unbounded"/>
             </xs:sequence>
             <xs:attribute name="OID" type="string" use="required"/>
          </xs:complexType>
       </xs:element>

   <!--  COMMENT:   ItemGroupDef  -->

      <xs:attribute name="Label" type="string"/>
      <xs:attribute name="Structure" type="string"/>
      <xs:attribute name="DomainKeys" type="string"/>
      <xs:attribute name="Class" type="string"/>
      <xs:attribute name="ArchiveLocationID" type="string"/>

   <!--  COMMENT:   ItemDef - Label attribute is also valid on ItemDef Element  -->

      <xs:attribute name="DisplayFormat" type="string"/>
      <xs:attribute name="ComputationMethodOID" type="string"/>

      <xs:element name="ValueListRef">
         <xs:complexType>
            <xs:attribute name="ValueListOID" type="string" use="required"/>
         </xs:complexType>
      </xs:element>

   <!--  COMMENT:   CodeListItem  -->

      <xs:attribute name="Rank" type="float"/>
</xs:schema>
```

- define-extension.xsd: The extension mechanism itself.
  File:

  define-extension.xsd

  Contents:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema targetNamespace="http://www.cdisc.org/ns/odm/v1.2"
           xmlns="http://www.cdisc.org/ns/odm/v1.2"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
           xmlns:def="http://www.cdisc.org/ns/def/v1.0"
           xmlns:odm="http://www.cdisc.org/ns/odm/v1.2"
           elementFormDefault="qualified" attributeFormDefault="unqualified">


    <xs:import namespace="http://www.cdisc.org/ns/def/v1.0"
schemaLocation="http://www.cdisc.org/schema/def/v1.0/define-ns.xsd"/>


    <xs:redefine schemaLocation="http://www.cdisc.org/schema/odm/v1.2.1/ODM1-2-1.xsd">

    <!--
          MetaDataVersion
      -->
      <xs:attributeGroup name="MetaDataVersionAttributeExtension">
      <xs:attributeGroup  ref="MetaDataVersionAttributeExtension"/>
          <xs:attribute ref="def:DefineVersion"      use="required"/>
          <xs:attribute ref="def:StandardName"    use="required"/>
          <xs:attribute ref="def:StandardVersion" use="required"/>
      </xs:attributeGroup>

      <xs:group name="MetaDataVersionPreIncludeElementExtension">
            <xs:sequence>
              <xs:group ref="MetaDataVersionPreIncludeElementExtension"/>
              <xs:element ref="def:AnnotatedCRF" minOccurs="0" maxOccurs="unbounded"/>
              <xs:element ref="def:SupplementalDoc" minOccurs="0"
maxOccurs="unbounded"/>
              <xs:element ref="def:leaf" minOccurs="0" maxOccurs="unbounded"/>
              <xs:element ref="def:ComputationMethod" minOccurs="0"
maxOccurs="unbounded"/>
              <xs:element ref="def:ValueListDef" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
      </xs:group>



    <!--
          ItemGroupDef
      -->
      <xs:attributeGroup name="ItemGroupDefAttributeExtension">
      <xs:attributeGroup  ref="ItemGroupDefAttributeExtension"/>
          <xs:attribute ref="def:Label" use="required"/>
          <xs:attribute ref="def:Class"/>
          <xs:attribute ref="def:Structure"/>
          <xs:attribute ref="def:DomainKeys"/>
          <xs:attribute ref="def:ArchiveLocationID" use="required"/>
      </xs:attributeGroup>

      <xs:group name="ItemGroupDefElementExtension">
```

```
        <xs:sequence>
           <xs:group ref="ItemGroupDefElementExtension"/>
           <xs:element ref="def:leaf" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:group>



    <!--
          ItemDef
          Label attribute is also valid on ItemDef Element
      -->
      <xs:attributeGroup name="ItemDefAttributeExtension">
      <xs:attributeGroup  ref="ItemDefAttributeExtension"/>
          <xs:attribute ref="def:Label"/>
          <xs:attribute ref="def:DisplayFormat"/>
          <xs:attribute ref="def:ComputationMethodOID"/>
      </xs:attributeGroup>

      <xs:group name="ItemDefElementExtension">
          <xs:sequence>
              <xs:group ref="ItemDefElementExtension"/>
              <xs:element ref="def:ValueListRef" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
       </xs:group>



    <!--
          CodeListItem
      -->
      <xs:attributeGroup name="CodeListItemAttributeExtension">
      <xs:attributeGroup  ref="CodeListItemAttributeExtension"/>
          <xs:attribute ref="def:Rank"/>
      </xs:attributeGroup>


    </xs:redefine>


</xs:schema>
```

- define1-0-0.xsd: The application reference file.

File:

define1-0-0.xsd

Contents:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema targetNamespace="http://www.cdisc.org/ns/odm/v1.2"
           xmlns="http://www.cdisc.org/ns/odm/v1.2"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
           xmlns:def="http://www.cdisc.org/ns/def/v1.0"
           elementFormDefault="qualified" attributeFormDefault="unqualified">

    <!-- COMMENT: Import ODM foundation schemas  -->

    <xs:import namespace="http://www.w3.org/XML/1998/namespace"
schemaLocation="http://www.w3.org/2001/03/xml.xsd"/>
    <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
schemaLocation="http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd"/>

    <!-- COMMENT: Include here any vendor extensions  -->

    <xs:include schemaLocation="http://www.cdisc.org/schema/def/v1.0/define-extension.xsd"/>
</xs:schema>
```

5. An define.xml Example

5.1. The Example

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<ODM
  xmlns="http://www.cdisc.org/ns/odm/v1.2"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:def="http://www.cdisc.org/ns/def/v1.0"
  xsi:schemaLocation="http://www.cdisc.org/ns/odm/v1.2 define1-0-0.xsd"
  FileOID="Study1234"
  ODMVersion="1.2"
  FileType="Snapshot"
  CreationDateTime="2004-07-28T12:34:13-06:00">
<Study OID="Study1234">
  <GlobalVariables>
    <StudyName>1234</StudyName>
    <StudyDescription>The description of study 1234</StudyDescription>
    <ProtocolName>Study 1234</ProtocolName>
  </GlobalVariables>
<MetaDataVersion OID="CDISC.SDTM.3.1.0"
    Name="CDISC SDTM Data for Study 1234"
    Description="Study 1234, Data Definitions"
    def:DefineVersion="1.0.0"
    def:StandardName="CDISC SDTM"
    def:StandardVersion="3.1.0">
<def:AnnotatedCRF>
    <def:DocumentRef leafID="blankcrf"/>
</def:AnnotatedCRF>
<def:SupplementalDoc>
    <def:DocumentRef leafID="SupplementalDataDefinitions"/>
</def:SupplementalDoc>
<def:leaf ID="blankcrf" xlink:href="blankcrf.pdf">
    <def:title>Annotated Case Report Form</def:title>
</def:leaf>
<def:leaf ID="SupplementalDataDefinitions"
     xlink:href="SupplementalDataDefinitions.pdf">
    <def:title>Supplemental Data Definitions</def:title>
</def:leaf>
    <def:ValueListDef OID="ValueList.VS.VSTESTCD">
      <ItemRef ItemOID="VS.VSTESTCD.FRAME" OrderNumber="1" Mandatory="No"/>
    </def:ValueListDef>
```

```
<!-- ********************************************************** -->
<!-- ItemGroupDef (domain) information Section (for the DM domain) -->
<!-- ********************************************************** -->
<ItemGroupDef OID="dm"
   Name="DM"
   Repeating="No"
   IsReferenceData="No"
   Purpose="Tabulation"
   def:Label="Demographics"
   def:Structure="One Record per Subject"
   def:DomainKeys="USUBID, DMSEQ"
   def:Class="Demographics"
   def:ArchiveLocationID="ArchiveLocation.dm">

   <!-- ***************************************************** -->
   <!-- Each variable is listed here for this domain          -->
   <!-- ***************************************************** -->
      <ItemRef ItemOID="DOMAIN" OrderNumber="1" Mandatory="Yes" Role="IDENTIFIER"/>
      <ItemRef ItemOID="USUBJID" OrderNumber="2" Mandatory="Yes" Role="IDENTIFIER"/>
      <ItemRef ItemOID="dm.AGE" OrderNumber="3" Mandatory="Yes" Role="QUALIFIER"/>
      <!-- COMMENT: Additional variables would be listed here -->
   <def:leaf ID="ArchiveLocation.dm" xlink:href="dm.xpt">
   <def:title>crt/datasets/1234/dm.xpt</def:title>
  </def:leaf>
</ItemGroupDef>
<def:ComputationMethod OID="SEQNO">Use SQL rownum function</def:ComputationMethod>

<!-- ********************************************************** -->
<!-- ItemGroupDef (domain) information Section (for the AE domain) -->
<!-- ********************************************************** -->
<ItemGroupDef OID="docroot.IG.AE"
  Name="AE"
  Repeating="Yes"
  IsReferenceData="No"
  Purpose="Tabulation"
  def:Label="Adverse Events"
  def:Structure="One record per subject per event"
  def:DomainKeys="USUBJID, AETERM, AESEQ"
  def:Class="Events"
  def:ArchiveLocationID="ArchiveLocation.ae">
      <ItemRef ItemOID="DOMAIN" OrderNumber="1" Mandatory="Yes" Role="Identifier"/>
      <ItemRef ItemOID="USUBJID" OrderNumber="2" Mandatory="Yes" Role="Identifier"/>
      <ItemRef ItemOID="AESEQ" OrderNumber="3" Mandatory="No" Role="Identifier"/>
      <ItemRef ItemOID="AETERM" OrderNumber="4" Mandatory="Yes" Role=" Topic"/>
      <ItemRef ItemOID="AEDECODE" OrderNumber="5" Mandatory="Yes" Role="Result Qualifier"/>
      <!-- COMMENT: Additional variables would be listed here -->
   <def:leaf ID="ArchiveLocation.ae" xlink:href="ae.xpt">
   <def:title>ae.xpt</def:title>
   </def:leaf>
</ItemGroupDef>
```

```
<!-- ********************************************************** -->
<!-- ItemGroupDef (domain) information Section (for the VS domain) -->
<!-- ********************************************************** -->
 <ItemGroupDef OID=" VS"
   Name="VS"
   Repeating="Yes"
   IsReferenceData="No"
   Purpose="Tabulation"
   def:Label="Vital Signs"
   def:Structure="One record per subject per vital sign"
   def:DomainKeys="USUBJID, VSTESTCD"
   def:Class="Findings"
   def:ArchiveLocationID="Location.VS">
       <ItemRef ItemOID="DOMAIN" OrderNumber="1" Mandatory="Yes" Role="Identifier"/>
       <ItemRef ItemOID="USUBJID" OrderNumber="2" Mandatory="Yes" Role="Identifier"/>
       <ItemRef ItemOID="VSSEQ" OrderNumber="3" Mandatory="Yes" Role="Identifier"/>
       <ItemRef ItemOID="VSTESTCD" OrderNumber="4" Mandatory="Yes" Role="Topic"/>
       <!-- COMMENT: Additional variables would be listed here -->
   <def:leaf def:ID="Location.VS" xlink:href="VS.xpt">
     <def:title>crt/datasets/1234/VS.xpt</def:title>
   </def:leaf>
 </ItemGroupDef>

<!-- COMMENT: Additional domains would be listed here -->
```

```
<!-- ************************************************************ -->
<!-- The details of each variable (all variables) would be listed here -->
<!-- ************************************************************ -->
<ItemDef OID="DOMAIN"
   Name="DOMAIN"
   DataType="text"
   Length="2"
   Origin="CRF Page"
   Comment="Two-character abbreviation for the domain"
   def:Label="DOMAIN ABBREVIATION">
   <CodeListRef CodeListOID="CodeList.Domain"/>
</ItemDef>

<ItemDef OID="USUBJID" Name="USUBJID"
   DataType="text"     Length="20"
   Origin="CRF Page"
   Comment="Unique subject identifier within the submission."
   def:Label="UNIQUE SUBJECT IDENTIFIER">
</ItemDef>

<ItemDef OID="dm.AGE"  Name="AGE"
   DataType="float"    Length="8"   SignificantDigits="4"
   Origin="CRF Page"   Comment="The subjects AGE at screening."
   def:Label="AGE IN AGEU AT REFERENCE Datetime"
   def:DisplayFormat="3.0">
</ItemDef>

<ItemDef OID="AESEQ"   Name="AESEQ"
   DataType="integer"  Length="5"
   Origin="Derived"
   def:Label="Used to Sequence AEs within subjects"
   def:ComputationMethodOID="SEQNO">
</ItemDef>

<ItemDef OID="AEDECOD" Name="AEDECOD"
   DataType="text"     Length="80"
   Origin="Derived"
   def:Label="MedDRA code for AETERM">
   <CodeListReF CodeListOID="docroot.CL.MEDDRA"/>
</ItemDef>

<ItemDef OID="VSSEQ"   Name="VSSEQ"
   DataType="integer"  Length="8"
   Origin="CRF Page"
   Comment="SEQUENCE NUMBER"
   def:Label="SEQUENCE NUMBER" >
</ItemDef>

<ItemDef OID="VSTESTCD" Name="VSTESTCD"
   DataType="text"     Length="8"
   Origin="CRF Page"   Comment="Vital Signs CRF Page 4"
   def:Label="VITAL SIGNS TEST SHORT NAME" >
   <def:ValueListRef ValueListOID="ValueList.VS.VSTESTCD"/>
</ItemDef>

<ItemDef OID="VS.VSTESTCD.FRAME" Name="FRAME"
   DataType="integer"  Length="8"
   Origin="CRF Page"
   def:Label="Body Frame Size">
<CodeListRef CodeOID="Code.FRAME"/>
</ItemDef>
```

```
  <!-- ******************************************************* -->
  <!--  All Decodes would be listed here                       -->
  <!-- ******************************************************* -->
  <CodeList OID="CodeList.Domain"
       Name="Domains"
       DataType="text">

       <CodeListItem CodedValue="DM">
          <Decode>
             <TranslatedText xml:lang="en">Demographics</TranslatedText>
          </Decode>
       </CodeListItem>

       <CodeListItem CodedValue="AE">
          <Decode>
             <TranslatedText xml:lang="en">Adverse Event</TranslatedText>
          </Decode>
       </CodeListItem>

    </CodeList>

    <CodeList OID="CodeList.CL.MEDDRA"
       Name="MedDRA"
       DataType="text">
       <ExternalCodeList Dictionary="MedDRA" Version="7.1"/>
    </CodeList>

    <CodeList OID="Code.FRAME" Name="FRAME" DataType="integer">

      <CodeListItem CodedValue="1">
        <Decode>
          <TranslatedText xml:lang="en">Small</TranslatedText>
        </Decode>
      </CodeListItem>

      <CodeListItem CodedValue="2">
        <Decode>
          <TranslatedText xml:lang="en">Medium</TranslatedText>
        </Decode>
      </CodeListItem>

      <CodeListItem CodedValue="3">
        <Decode>
          <TranslatedText xml:lang="en">Large</TranslatedText>
        </Decode>
      </CodeListItem>

      <CodeListItem CodedValue="4">
        <Decode>
          <TranslatedText xml:lang="en">Extra large</TranslatedText>
        </Decode>
      </CodeListItem>

    </CodeList>
  </MetaDataVersion>
 </Study>
</ODM>
```

5.2. *Global Ordering of Elements*

Following the order of elements is apart of conformance to the CRT DD specification as stated in Section 2.3, therefore, all of the elements are listed below in their correct order for your reference.

5.2.1. Global Element Order

1. ODM
    a. Study
        i. GlobalVariables
            1. StudyName
            2. StudyDescription
            3. ProtocolName
        ii. MetaDataVersion
            1. def:AnnotatedCRF
                a. def:DocumentRef
            2. def:SupplementalDoc
                a. def:DocumentRef
            3. def:leaf
                a. def:title
            4. def:ComputationMethod
            5. def:ValueListDef
                a. ItemRef
            6. Include
            7. ItemGroupDef
                a. ItemRef
                b. def:leaf
                    i. def:title
                c. def:ValueListRef
            8. ItemDef
                a. CodeListRef
            9. CodeList
                a. CodeListItem
                    i. Decode
                        1. TranslatedText
                b. ExternalCodeList

6. The define.xml Team

The define.xml team was reestablished in May of 2003 to develop a machine-readable data definition document, supporting toolset to replace the PDF version of the Data Definition Document, and to facilitate the data exchange between sponsors and the FDA in support of regulatory filings.

Role	Name	Company
Team Lead	William J. Qubeck, IV	Pfizer, Inc.
Team Co-Leads	Sally Cassells	Lincoln Technologies, Inc.
	Michael Palmer	Zurich Biostatistics, Inc.
	Joyce Hernandez	IBM Life Sciences
Team Participants	Jozef Aerts	XML4Pharma
	Alex Bajamonde	Genentech, Inc.
	Neeru Bhardwaj	Sanofi-Synthélabo
	Kevin Burges	Formedix
	Dave Christiansen	Christiansen Consulting
	Anthony Friebel	SAS Institute Inc.
	William Friggle	Sanofi-Synthélabo
	Thomas Guinter	Octagon Research Solutions, Inc.
	Raymond Heimbuch	Novartis Corp.
	Wayne Kubick	Lincoln Technologies, Inc.
	Sandy Lei	Johnson & Johnson Pharmaceutical, L.L.C.
	Milan Shah	Johnson & Johnson Pharmaceutical, L.L.C.
	Gary Walker	Quintiles Transnational Corp.
FDA Participant	Steve Wilson	Center for Drug Evaluation and Research (CDER), U. S. Food and Drug Administration
Special Thanks	CDISC ODM Team	