XML Schema Validation for Define.xml

Date: 30 November 2009

Version: 1.0

Document History

Issue	Author	Date	Description
0.1	CDISC XML Technology Team	30 September 2009	First draft.
1.0	CDISC XML Technology Team	30 November 2009	Initial release.

CDISC, Inc.
15907 Two Rivers Cove, Austin, Texas 78717
http://www.cdisc.org

Table of Contents

1 Introduction

This white paper provides guidance on validating *define.xml* documents against the *define.xml* XML schemas. The goal is to foster more consistent validation results in order to facilitate regulatory submissions and interchange of *define.xml* documents.

The scope of this white paper is limited to providing guidance on *define.xml* schema validation. It does not provide guidance on other important areas of *define.xml* validity including:

- Full compliance with the published define v1.0 specification
- Accuracy of content versus the SDTM datasets and other related documents
- Any aspects of quality beyond what XML schema can define

This white paper proposes practices and tools to improve *define.xml* schema validation. This document does not include any *define.xml* specifications or recommendations for creating *define.xml* content.

For information on the *define.xml* standard, officially known as the **Case Report Tabulations Data Definition Specification (CRT-DDS)**, please visit the CDISC web site at http://www.cdisc.org/content1057for v1.0 of the standard. The web site documentation for *define.xml* includes:

- The *define.xml* specification document
- The *define.xml* schema files
- Example *define.xml* instance files
- **SchemaFiles.zip** – bundle of the *define.xml* schema files
- **ExampleFiles.zip** - an example *define.xml* file that includes the corresponding datasets and PDF documents.
- **Define_validation.zip** – a newly posted file that packages 3 define examples with copies of the schemas to facilitate testing *define.xml* validation
- Other files include a style sheet and zipped packages of the above files.

2 The Define.xml Standard

The specification document available on the CDISC web site provides the normative definition of the standard. It includes all the necessary information for developing compliant *define.xml* files that facilitate the exchange and interpretation of the associated SDTM datasets.

The XML schema used to define the expected structure for *define.xml* is based on an extension to the CDISC Operational Data Model (ODM) version 1.2.1. The *define.xml* schema is composed of a number of separate schema documents including:

- **define1-0-0.xsd:** The *define.xml* schema version 1.0. The main schema that glues the extensions and the ODM together to create the full *define.xml* definition. Most applications implementing *define.xml* should only directly reference this schema file.
- **define-extension.xsd:** This schema identifies the location for the *define.xml* extensions within the ODM.
- **define-ns.xsd:** This schema defines the elements and attributes that are included in the *define.xml* extensions to ODM.
- **ODM1-2-1.xsd:** Main schema for ODM version 1.2.1.
- **ODM1-2-1-foundation.xsd:** Defines the elements, attributes and structure of the base ODM schema.

All the schema files listed above are necessary to validate a *define.xml* instance. To include a reference to these schemas in a *define.xml* instance, it is only necessary to reference the **Define1-0-0.xsd** schema file.

The CDISC authored *define.xml* schema files reference standard schema documents published by the W3C standards development organization. These files are also necessary to validate *define.xml* instance documents:

- **xml.xsd:** Schema that describes the XML namespace in a form suitable for import by other schema documents
- **xmldsig-core-schema.xsd:** Schema that specifies the XML syntax and processing rules for creating and representing digital signatures
- **xlink.xsd:** Schema that defines the XML Linking Language (XLink) used to create and describe links between resources
- **datatypes.dtd:** DTD that supports the datatype definitions in the XML Schema recommendation

The newly added **define_validation.zip** file was created to facilitate testing *define.xml* validation. This zip file includes a number of different files:

- 3 *define.xml* instance example files
- 5 CDISC schema files for *define.xml* and ODM
- 3 W3C xml standard schemas and 2 standard DTDs
- Readme.txt for validation hints and instructions

All the schemas listed above are used to validate the *define.xml* example documents. Successfully validating the *define.xml* examples typically indicates that the same tools and processes should be capable of validating your own *define.xml* documents (assuming, of course, that the content of your *define.xml* file is valid).

3 XML Schema Validation

Validating a *define.xml* instance means executing a process to ensure that the XML document follows the structure and content rules specified in the define schemas. Processing a *define.xml* file is difficult or impossible for the receiving party when the file does not conform to the expected structure. For this reason, *define.xml* documents submitted to a regulatory agency or exchanged with an external partner are typically validated against the *define.xml* schema prior to processing their contents.

In general, schemas are used to validate:

- **Data completeness.** Schema validation checks to ensure all information required by the schema is present in the *define.xml* instance.
- **Data structure.** Schema validation checks to ensure that the basic structure of the elements and attributes in the *define.xml* instance matches the schema.
- **Data correctness.** Schema validation indicates that the data conforms to the rules of the schema. However, it does not guarantee that the information is meaningful from the consumer's perspective.

XML Schema is a standard, or recommendation, published by the World Wide Web Consortium (W3C). For more of the details on the XML Schema Recommendation visit the W3C Web site at http://www.w3.org/XML/Schema. Read http://www.w3.org/TR/xmlschema-0/#conformance for a description of schema validation.

Schema validation determines whether a *define.xml* instance conforms to all of the constraints described in the schemas. However, the XML Schema Recommendation does not dictate how XML schema tools must validate an XML document.

XML schema validation is a first step towards verifying that a *define.xml* instance matches the published standard. Since the schema cannot represent all the rules and requirements documented in the specification, additional checks should be executed to ensure compliance with the complete specification. There are a number of tools provided by industry vendors to perform this level of conformance checking, including:

- Formedix **Origin Submission Modeller** http://www.formedix.com/index.php
- SAS **Clinical Standards Toolkit** http://www.sas.com/industry/pharma/cdisc/ and **Clinical Data Integration 2.1** http://www.sas.com/industry/pharma/cdi/index.html
- Phase Forward **WebSDM** http://www.phaseforward.com/products/cdisc/
- XML4Pharma **Define.xml Checker** http://www.xml4pharma.com/CDISC_Products/index.html#definechecker
- OpenCDISC **Validator** http://www.opencdisc.org/

4 Challenges Validating Define.xml

Since the XML schema recommendation does not dictate how to implement schema validation software, not all tools will validate a *define.xml* document the same way. Tools may:

- Interpret the recommendation differently
- Use different implementation strategies that cause validation outcomes to vary.

In general, the ***import*** and ***redefine*** elements seem to be the source of much of the variation among schema validation software implementations. *Define.xml*'s use of ***import*** and ***redefine*** has been peer reviewed for compliance with the XML Schema Recommendation and is supported by most schema validation software. See http://www.w3schools.com/Schema/schema_elements_ref.asp for definitions of ***import*** and ***redefine***.

Unfortunately, schema validation is not always as simple as getting a **Pass** or **Fail** outcome. Sometimes an XML document will **Pass** validation, but there will be warning messages. Warning messages must be evaluated on a case-by-case basis for relevance.

It is also important to note that attaining a **Pass** outcome does not always mean that all content in the *define.xml* file was evaluated against schema definitions. Some XML parsers, such as XSV or MSXML 4.0 perform lax validation. This means that the validating software reports errors for the elements and attributes that are declared in the define schemas, but does not report errors for content not found in the schema.

Generally speaking, if a *define.xml* instance **Passes** validation, it should be processable by the tools that are expected to interpret the associated SDTM datasets. In practice, XML schema validation tools produce relatively consistent validation results. However, it should be noted that some tools are not capable of validating a conforming *define.xml* instance.

5 Practices for Improving Define.xml Validation Success

The following *define.xml* validation practices are recommended:

1. **Use a tool that has successfully validated *define.xml*.**
    a. Check the list of tools tested against the examples in **Tools Table**
    b. Avoid use of the **Altova XMLSpy** product for *define.xml* validation. Altova maintains an interpretation of the schema elements ***import*** and ***redefine*** that is not compatible with *define.xml* validation. The W3C XML Schema working group has concluded that there is disagreement regarding how to interpret this aspect of the schema recommendation.
2. **Use local copies of the schemas.**
    a. Creating a local schema repository is recommended
    b. Referencing the publicly available schemas on the CDISC web site can be convenient, but issues connecting to the site, security concerns, or software that ignores **xsi:schemaLocation** may cause less consistent results than using local versions.
3. **Referencing the define1-0-0 schema in *define.xml* documents**
    a. Providing the location of the local define schema to the validating tool at validation time is the recommended approach.
    b. Setting the value for **xsi:schemaLocation** to reference a local schema in the same directory as the *define.xml* instance
4. **Run tests using the example *define.xml* files.**
    a. Test your validation tools and process using the *define.xml* examples
    b. Test your own *define.xml* file using the same tools and processes used to validate the examples
5. **Post questions on the CDISC XML Validation forum**
    a. Post questions here http://bbs.cdisc.org/bbs/category-view.asp
    b. Review existing posts and responses before posting a new question.

These practices are explained in more detail in the sections below.

5.1 Use a Tool That Has Successfully Validated Define.xml

As part of the effort to create this white paper, members of the CDISC XML Technologies Team ran schema validation tests using the *define.xml* example files and popular XML tools. The list of XML tools evaluated is not exhaustive, and there are other tools that will successfully validate *define.xml*. The **Tools Table** below shows high-level results from the XML Technologies Team's testing efforts.

5.1.1.1 Tools Table – Define.xml schema validation test results

Tool	XML Parser	Result	Notes
Oxygen 10.3	Xerces-J SaxonSA	PASS	Oxygen uses the Xerces parser by default, but has the option of setting alternative parsers. Not all available parsers successfully validate the *define.xml* examples.
	MSXML4.0	FAIL	
Eclipse 3.3.1.1	Xerces-J 2.8.0	PASS	Retargeted the **xsi:schemaLocation** to reference local copies of the schemas
XMLExchanger	Xerces-J	PASS	

	2.7.1		
XMLBlueprint 6.2.1111	Xerces2-J Libxml2	PASS	XMLBlueprint provides the option of 4 different parsers. Not all available parsers successfully validate the *define.xml* examples.
	MSXML4.0 MSV	FAIL	
Liquid XML		FAIL	Error: "The 'http://www.w3.org/2000/09/xmldsig#:Signature' element is not declared."
Validome.org		PASS	Online xml validation. Requires access to online copies of the schemas via a URL
ANT	JAXP	PASS	
Define Validator (WebSDM command line XML validator)	Xerces 2.9.0	PASS	Now available for free download - see the **Appendix** for the URL
XMLSpy	AltovaXML	FAIL	Altova has acknowledged that the AltovaXML parser will not validate *define.xml*.
Webdata/xsv	XSV	PASS	Online validator. Add the URL of the example define.xml files in the Address box
XMLStarlet 1.0.1	Libxml 2.7.3	PASS	Command-line tool that makes use of the GNOME XML library.
SAS	Xerces 2.7.1	PASS	Tests run with the SAS XML Mapper. The XML Mapper will validate *define.xml*, but will not necessarily show all errors, or explain the errors in full detail.

To share experiences using different tools, or to ask questions about the above tools, please post to the CDISC XML Validation Forum at http://bbs.cdisc.org/bbs/category-view.asp.

The **Appendix** lists the URLs for the tools tested, as well as the URLs for a number of the XML parsers used by the tools.

5.2 Use Local Copies of the Schemas

Using local copies of the schemas rather than referencing schemas located at a URL on the Web is recommended. This increases the probability that the validating software can find and access the appropriate schema files.

Creating a local schema repository for the schema files used to validate *define.xml* is a recommended best practice. Creating a local schema repository is as simple as creating a folder structure on your hard drive to store the schema files. A few hints on how to do this:

- Use namespace names to provide guidance on naming the local folders
- Use version numbers in the structure such that the repository can be updated with new schema versions as they are released
- Consider other CDISC XML technology schemas you may want to include, such as ODM 1-3, ODM 1-3-1, and Lab 1-0-1
- Using a shared drive on a local area network is acceptable, but access to the schemas will not be possible when you are not connected to the network

It is also possible to reference online versions of the schemas. These are publicly available from CDISC at http://www.cdisc.org/schema/def/v1.0/define1-0-0.xsd. This can be especially convenient when sending the *define.xml* file to another organization. However, local copies of the schemas are more reliably available, and local schema repositories maintained by the sender and receiver are recommended.

5.3 Referencing the Define1-0-0 Schema in Define.xml Documents

Many tools enable you to link the schema to the *define.xml* file at validation time, and this method is generally considered to be a robust option for referencing the schema. This approach is often preferred because the consuming application is typically better suited to establishing the location of your local schema repository.

Another approach is to set the value of the **xsi:schemaLocation** attribute within the *define.xml* file itself. This attribute value allows you to tie the *define.xml* document to the **define-1-0-0.xsd** schema. This link is not mandatory, but with many validators it does help the tool to locate the schema. The W3C Schema Recommendation considers the **xsi:schemaLocation** value as a hint to the validating software on where to find the schema.

In general, validation tools will use a handful of possible strategies to identify and locate the *define.xml* schema for validation purposes:

- Use the **xsi:schemaLocation** attribute value to find the schema. Not all tools will make use of this hint.
- Enable the user to set the location of the schema within the validating software at the time of validation processing.
- The validating software may dereference the namespace to retrieve the schema.
- Use logic built-in to the application to determine the location of the define schema.

Not all XML parsers employ these strategies in the same way. For example Xerces, a broadly used XML parser maintained by the Apache Software Foundation uses the information in **xsi:schemaLocation** to locate the *define.xml* schema. On the other hand, the MSXML 4.0 parser does not.

The recommended **xsi:schemaLocation** setting and the one used in the *define.xml* examples is shown in **Example 1** below:

Example 1 (recommended setting for the sender):

xsi:schemaLocation="http://www.cdisc.org/ns/odm/v1.2 define1-0-0.xsd"

The value for **xsi:schemaLocation** is actually 2 values separated by a whitespace. The first value is the namespace name and the second value is the URI for the schema location. Here **xsi:schemaLocation** relates the namespace http://www.cdisc.org/ns/odm/v1.2 to the schema location **define1-0-0.xsd**. In this case the URI for the **xsi:schemaLocation** is simply the schema file name, and this tells the validating software to look for the named schema file in the same folder as the *define.xml* instance document.

To reference a schema in a different folder, change the location portion of the attribute value. The following shows an example of accessing a local schema using a full path:

Example 2:

xsi:schemaLocation="http://www.cdisc.org/ns/odm/v1.2 file:///c:/xml/ex1/define1-0-0.xsd"

Relative paths to schema files also work, as shown in Example 3:

Example 3:

xsi:schemaLocation="http://www.cdisc.org/ns/odm/v1.2 ../util/define1-0-0.xsd"

However, if this *define.xml* will be sent to an external agency or partner, including a path to a local schema on your hard drive is not portable. This path will generally not be valid for the receiver. Senders are recommended to set **xsi:schemaLocation** to the value shown in **Example 1**.

To reference an online version of the define1-0-0 schema, a URI to a publicly available version of the define standard can be used, as in the following example:

Example 4:

xsi:schemaLocation="http://www.cdisc.org/ns/odm/v1.2
http://www.cdisc.org/schema/def/v1.0/define1-0-0.xsd"

Example 4 relies on the validator's ability to access the schema at the URL listed. This is a limitation. However, eliminating the need for local copies of the schemas can be a benefit, and is certainly a convenience. As previously noted, using local copies of the schemas is the most reliable method. When sending a *define.xml* file to a third party, following the recommendation shown in **Example 1** is preferred. However, it is acceptable for the receiver to set the validating tool to reference the online version of the schema, as shown in **Example 4**, if this is their preference.

The URL listed as the namespace leads some people to believe that they can use a browser to retrieve the schema like a Web page. In fact, using the ODM namespace URI http://www.cdisc.org/ns/odm/v1.2 results in a 404 page not found error. There is significant debate on best practices for dereferencing the namespace URI, and generally speaking users should not rely on it for schema validation.

5.4 Run Tests Using the Example Define.xml Files

Before validating production *define.xml* files, it is worthwhile to test your validation tools and processes using the examples found in the **define_validation.zip**. Successful validation of the examples typically indicates that the same tool and process should successfully validate your own conforming *define.xml* files. If you successfully validate the examples and validation fails on your *define.xml* files, that indicates likely problems in your *define.xml* file content.

To validate the example define files follow these basic steps:

1. Create a folder to hold the define examples
2. Download the **define_validation.zip** file from http://www.cdisc.org/content1057
3. Extract the **define_validation.zip** files into your new folder
4. If your schema validation tool supports it, set the tool to use the **define1-0-0.xsd** in the local directory for the first example file, **define_ex1.xml**
5. Execute the validation feature of your XML tool to validate the schema. If you've recently validated a *define.xml* file using another schema, it may be necessary to use the tool to reset the cache and re-validate.
6. If it doesn't pass validation, check for your tool in our **Tools Table** list. If your tool has been shown not to support *define.xml*, you will be better served by using a tool that has successfully validated the examples.
7. If you're still having trouble, post your experience on the CDISC XML Validation forum to get help http://bbs.cdisc.org/bbs/category-view.asp.

5.5 Post Questions on the CDISC XML Validation Forum

If you're having difficulty validating the *define.xml* examples or your own *define.xml* files, post your question on the CDISC XML Validation forum. This forum is monitored by many of the CDISC *define.xml* experts, and posting a question is an opportunity to get input from them. Please include detailed information on the tools used and the output received, as well as the results of your attempt to validate the *define.xml* example files.

The XML Validation forum is also the place to share your experiences, successful or otherwise, using various XML tools for validating *define.xml* or ODM or any other CDISC XML standard. Please review existing posts to determine if your question has already been answered.

The XML Validation forum is only intended for questions relating to XML Schema validation on the CDISC XML standards. Other forums exist to answer questions about the *define.xml* specification or the SDTM standard.

6 Appendix

6.1 Web Sites for the XML Tools Evaluated

- DefineValidator https://www.phaseforward.com/products/cdisc/
- Oxygen XML Editor http://www.oxygenxml.com
- Eclipse http://www.eclipse.org/
- Cladonia XMLExchanger http://www.exchangerxml.com
- Liquid XML http://www.liquid-technologies.com/
- Validome http://www.validome.org
- Webdata/XSV http://www.w3.org/2001/03/webdata/xsv
- SAS http://www.sas.com
- XML Blueprint http://www.xmlblueprint.com/
- Liquid XML http://www.liquid-technologies.com/
- ANT http://ant.apache.org/manual/OptionalTasks/xmlvalidate.html
- XMLSpy http://www.altova.com/
- XMLStarlet http://xmlstar.sourceforge.net/
- XMLExchanger http://www.exchangerxml.com/

6.2 Web Sites for XML Parsers

- Xerces http://xerces.apache.org/
- Xerces2-Java http://xerces.apache.org/xerces2-j/index.html
- Saxon http://www.saxonica.com/
- JAXP https://jaxp.dev.java.net/
- MSXML 4.0 http://msdn.microsoft.com/xml/default.aspx
- Libxml2 http://xmlsoft.org/
- MSV - Sun Multi-Schema Validator https://msv.dev.java.net
- AltovaXML http://www.altova.com