HL7_SPEC_VALUESETDEF_R1_STU_2016JUN

HL7 Specification: Characteristics of a Formal Value Set Definition, Release 1

Standard for Trial Use

June 2016

Publication of this standard for trial use and comment has been approved by Health Level Seven International (HL7). This standard is not an accredited American National Standard. The comment period for trial use of this standard shall end 18 months from the date of publication. Suggestions for revision should be submitted at http://www.hl7.org/dstucomments/index.cfm.

Following this 18 month evaluation period, this standard, revised as necessary, will be submitted to a normative ballot in preparation for approval by ANSI as an American National Standard. Implementations of this trial use standard shall be viable throughout the normative ballot process and for up to six months after publication of the relevant normative standard.

IMPORTANT NOTES:

HL7 licenses its standards and select IP free of charge. **If you did not acquire a free license from HL7 for this document,** you are not authorized to access or make any use of it. To obtain a free license, please visit http://www.HL7.org/implement/standards/index.cfm.

If you are the individual that obtained the license for this HL7 Standard, specification or other freely licensed work (in each and every instance "Specified Material"), the following describes the permitted uses of the Material.

A. HL7 INDIVIDUAL, STUDENT AND HEALTH PROFESSIONAL MEMBERS, who register and agree to the terms of HL7's license, are authorized, without additional charge, to read, and to use Specified Material to develop and sell products and services that implement, but do not directly incorporate, the Specified Material in whole or in part without paying license fees to HL7.

INDIVIDUAL, STUDENT AND HEALTH PROFESSIONAL MEMBERS wishing to incorporate additional items of Special Material in whole or part, into products and services, or to enjoy additional authorizations granted to HL7 ORGANIZATIONAL MEMBERS as noted below, must become ORGANIZATIONAL MEMBERS of HL7.

B. HL7 ORGANIZATION MEMBERS, who register and agree to the terms of HL7's License, are authorized, without additional charge, on a perpetual (except as provided for in the full license terms governing the Material), non-exclusive and worldwide basis, the right to (a) download, copy (for internal purposes only) and share this Material with your employees and consultants for study purposes, and (b) utilize the Material for the purpose of developing, making, having made, using, marketing, importing, offering to sell or license, and selling or licensing, and to otherwise distribute, Compliant Products, in all cases subject to the conditions set forth in this Agreement and any relevant patent and other intellectual property rights of third parties (which may include members of HL7). No other license, sublicense, or other rights of any kind are granted under this Agreement.

C. NON-MEMBERS, who register and agree to the terms of HL7's IP policy for Specified Material, are authorized, without additional charge, to read and use the Specified Material for evaluating whether to implement, or in implementing, the Specified Material, and to use Specified Material to develop and sell products and services that implement, but do not directly incorporate, the Specified Material in whole or in part.

NON-MEMBERS wishing to incorporate additional items of Specified Material in whole or part, into products and services, or to enjoy the additional authorizations granted to HL7 ORGANIZATIONAL MEMBERS, as noted above, must become ORGANIZATIONAL MEMBERS of HL7.

Please see http://www.HL7.org/legal/ippolicy.cfm for the full license terms governing the Material.

Ownership. Licensee agrees and acknowledges that **HL7 owns** all right, title, and interest, in and to the Trademark. Licensee shall **take no action contrary to, or inconsistent with**, the foregoing.

Licensee agrees and acknowledges that HL7 may not own all right, title, and interest, in and to the Materials and that the Materials may contain and/or reference intellectual property owned by third parties ("Third Party IP"). Acceptance of these License Terms does not grant Licensee any rights with respect to Third Party IP. Licensee alone is responsible for identifying and obtaining any necessary licenses or authorizations to utilize Third Party IP in connection with the Materials or otherwise. Any actions, claims or suits brought by a third party resulting from a breach of any Third Party IP right by the Licensee remains the Licensee's liability.

Following is a non-exhaustive list of third-party terminologies that may require a separate license:

Terminology	**Owner/Contact**
Current Procedures Terminology (CPT) code set	American Medical Association http://www.ama-assn.org/ama/pub/physician-resources/solutions-managing-your-practice/coding-billing-insurance/cpt/cpt-products-services/licensing.page?
SNOMED CT	International Healthcare Terminology Standards Development Organization (IHTSDO) http://www.ihtsdo.org/snomed-ct/get-snomed-ct or info@ihtsdo.org
Logical Observation Identifiers Names & Codes (LOINC)	Regenstrief Institute
International Classification of Diseases (ICD) codes	World Health Organization (WHO)
NUCC Health Care Provider Taxonomy code set	American Medical Association. Please see 222.nucc.org. AMA licensing contact: 312-464-5022 (AMA IP services)

Primary Editors: Robert McClure, MD, MD Partners, Inc.
rmcclure@mdpartners.com

W. Ted Klein, Klein Consulting, Inc.
ted@tklein.com

Additional Authors: George Beeler, Beeler Consulting LLC

Gay Dolin, MSN RN, Intelligent Medical Objects, Inc.
gdolin@imo-online.com

Grahame Grieve, Health Intersections
grahame@healthintersections.com.au

Russ Hamm, Lantana Consulting
russ.hamm@lantanagroup.com

Rob Hausam, MD, Hausam Consulting LLC
rrhausam@gmail.com

Wendy Huang, Canada Health Infoway Inc.
whuang@infoway-inforoute.ca

Julie James, Blue Wave Informatics LLP
julie_james@bluewaveinformatics.co.uk

Lloyd McKenzie, Gordon Point Informatics
lloyd@lmckenzie.com

Carmela Couderc, Intelligent Medical Objects, Inc
CCouderc@imo-online.com

Susan Barber, TN Department of Health
Susan.Barber@tn.gov

Table of Contents

Table of Figures

1 Notes to Readers

1.1 DataTypes used

This specification makes use of the ISO 21090 data types, though implementations are not restricted to using those data types. In some cases, a data type is marked with an asterisk (e.g., ST*). In this case, there is complex content that can be represented as the specified type, but there is a more complex underlying structure that some implementations may choose to represent in a more structured way. For example, a code, display name and description can be represented as a single string, even though they are discrete components. Because ISO 21090 doesn't provide a structure for conveying that combination of values and because the purpose is human readability, treating the structure as an ST for exchange purposes is appropriate.

Note that the datatypes shown in the diagrams are UML-datatype approximations of the ISO 21090 datatypes due to current tooling limitations. The intended datatypes are those noted in the prose.

2 Changes from Previous Release

This version has numerous changes based on the comments obtained during ballot review. In particular, this version now supports the use of alternative “expression syntaxes” to represent the Value Set Content Logical Definition (CLD). We continue to describe in the standard what we now call the An HL7 Value Set Definition Expression Syntax that is based on the HL7 Message Interchange Format (MIF) to serve as a ‘generalized” approach, but we also allow that other CLD representation syntaxes may be used.

3 Overview

3.1 Intended Audience

This specification is primarily intended for terminology system designers and individuals responsible for implementing standards that use terminology subsets. It is assumed the readers will be familiar with the terminology material in the current version of "**Core Principles and Properties of HL7 Version 3 Models."**[1] Some of the overview material will be important for less technically focused individuals to understand as they create terminology content for use in implemented systems.

Those individuals looking for the highlights should focus on the Value Set Definition Specification section as this provides the overall structure of a Value Set. . They should also review the details presented in the Content Logical Definition section. The section An HL7 Value Set Definition Expression Syntax provides a default set of functions that can be used (and *is* used for HL7 v3 Value Set definitions) to logically define the code content of a Value Set Expansion Code Set[2]. To understand how a Value Set Definition is not the same as a Value Set Expansion, see the sections Value Set Expansion File and also Relationship Between the Value Set Definition and Value Set Expansion Code Set.

Those readers particularly interested in understanding the set of functions used to define a Value Set Definition Content Logical Definition should study An HL7 Value Set Definition Expression Syntax and while this is generally useful for Value Set implementers, it is noted that other Value Set Definition Content Logical Definition formal syntaxes are now supported (see Content Expression).

3.2 In Scope

This document is a normative track specification that describes the data elements that formally define and characterize (describe) a Value Set. These include:

---

[1] http://www.hl7.org/implement/standards/product_brief.cfm?product_id=58

[2] See the Value Set Expansion File section for clarification on how a Value Set Expansion File contains the Value Set Expansion Code Set plus additional metadata and code attributes.

- Metadata used to identify and define a Value Set Definition (and includes an informative set of material on how to characterize a Value Set Expansion)
- The Functions that can be used to construct an "intentional" Value Set Definition Expression. This is described in the HL7 Value Set Definition Expression Syntax section. This expression is used in the Content Logical Definition.
- Elements to support Value Set Definition versioning.

The document also includes informative material describing "best practices" in the use of certain elements of the Value Set Definition, such as best approach versus allowed use of Concept Representations in Value Set Expansion Code Sets. It also includes an Informative section on Value Set Expansion File content.

3.3 Out of Scope

The following items have been declared explicitly out of scope for this initial release of this Standard:

- Implications of the use of this standard in evaluation of post-coordination concepts, even though support for inclusion of expressions is tacitly possible given the functionality described in the Content Logical Definition section.
- This standard does not describe an exchange model or syntax. A Value Set Definition exchange model and syntax standard may be the subject of future work.

The section Value Set Expansion File and all of the material in the Appendix are included as *Informative* material and are not a part of the Normative material of the specification.

3.4 Background

The definition of Value Sets is a required activity in completing the specification of most health information technology artifacts. To date, the approach for such definition has not been consistent within HL7 standards. Many of the required elements and approaches for Value Set Definition are embedded in existing HL7 artifacts (V2.x, V3, CDA and FHIR standards), but these are not applied or implemented consistently.

In HL7 normative standard "Core Principles and Properties of HL7 Version 3 Models" section 5.1.3, it is explicitly noted that "a Value Set is only persisted as its Value Set Definition, which is a machine-processable set of 1 or more formalisms that permit a specific collection of coded concepts at a given point in time to be reliably reproduced." Subsequent subsections elaborate on Value Set Expansions and the means of using the definition to produce them. The HL7 Common Terminology Services standards also describe various mechanisms and functions to produce Value Set Expansions from Value Set Definitions. However, to date there is not an explicit list of the precise data items

needed for the Value Set Definition expressed with the imprimatur of a normative standard. This STU is a major step to achieve that goal.

3.5 Need for this standard

Currently, an explicit list of all the fields in an HL7 Value Set Definition has surfaced only in the informatively-balloted MIF (Message Interchange Format), with a general description of the fields in the normative Core Principles specification. A more accessible and standardized description of the required elements is necessary to facilitate interoperability and sharing of Value Set Definitions across HL7 artifacts and the Health and Clinical Research IT communities at large.

This standard is intended to provide a formal specification of the data and metadata that are needed to formally define and describe a Value Set. Particular results include:

- Organizations within the Health and Clinical Research IT communities that implement this standard will have clearly defined and described Value Sets, including a clear description of the mechanism (functions to perform) to provide the expansion of the Value Set. This enables sharing of Value Set information accurately and unambiguously within and between organizations.
- Since Value Sets form a vital part of the artifacts needed to facilitate semantic interoperability, implementing this standard will support organizations in their efforts to achieve semantic interoperability in both internal and external communication.
- In addition, implementing this standard for Value Set Definition in applications such as repositories will support organizations in maintaining the meaning of the repository content over time, facilitating data aggregation and data analysis.

4 Controlled Vocabularies and Value Sets

Modern health care communications and data storage make heavy use of encoded information. There are many standards groups and bodies that have defined nomenclature and structures to handle such information.[3] In HL7, this is collectively referred to as "vocabulary". The HL7 published standards define several different types of objects that implement various characteristics of vocabulary. Whereas other elements of the HL7 standards are primarily concerned with information structures, vocabulary deals with content within those structures.

[3] Such as DICOM, WHO, LOINC, FDA, CDISC, and many others.

4.1 Concepts, Code Systems and Value Sets

A concept is a unitary mental representation of a real or abstract thing; an atomic unit of meaning. In structured vocabularies, these concepts are assigned codes. A code is a designation or identification by letters and or numerals that unambiguously identifies a concept in the vocabulary[4].

The context in which a concept is defined is called a code system. A code system is a collection of uniquely identifiable concepts with associated representations, designations, associations and meanings, published by a single organization or authority. Many code systems exist in the clinical domain, such as ICD-10, SNOMED CT, and LOINC, in addition to the code systems within HL7. Other code systems, such as ISO Country Codes, are widely used in many business areas.

A Value Set describes a collection of concepts drawn from one or more code systems grouped together for a specific purpose (e.g., orderable laboratory tests from LOINC). Note that the phrase "Value Set" usually is taken to mean both the Value Set Definition and the Value Set Expansion and for more detail on this, see section 5.1.2.

Value sets composed of codes from standard Code Systems are intended to provide common conceptual coverage for clinical concepts for exchange between systems. Local implementations may create constrained lists of string terms, such as may be employed in an EHR implementation. Constrained lists of strings are not considered Value Sets, but may be associated with a Value Set for exchange.

Typically implementation guides will reference Code Systems and Value Sets. This standard is primarily concerned with the elements that formalize the definition of Value Sets whose members are concepts derived from Code Systems.

4.2 Coded Data Types in HL7

The HL7 RIM is composed of classes with attributes that are data elements. A data element is a unit of data for which the definition, identification, representation and permissible values are specified. Every RIM data element has an HL7 data type. The data type defines the kind of values that can be contained by a data element. According to ISO 11404, a data type is "a set of distinct values, characterized by properties of those values and by

---

[4] This paraphrases the definition in HL7 Core Principles: Section 5.1.1 Concepts and Codes

operations on those values." A data type can be defined by intention, by extension or by a combination of these approaches. An intentional definition specifies the properties that the set of valid values must have (e.g., a definition that stipulates that a "string" is "an ordered collection of legible characters from a defined character set"). An extensional definition enumerates the values deemed valid (e.g., the assertion that the Boolean type consists of the values "true" and "false"). While extensional definitions are useful for coded attributes, almost all abstract data types are defined intentionally. Data type values stand for themselves; the value is all that counts. Neither identity nor state is defined for a data value. Data types have specific allowable values; but some may be implied, such as the data type of floating point numbers rather than being a list of values.

For those HL7 data elements whose data types are coded (Concept Descriptor, CD, or its specializations), the allowable values for the code property are coded concepts from code systems. These allowable values are specified in one or more Value Sets.

4.3 Terminology Binding

Most HL7 data elements do not permit all concepts from any code system to be used. A constraint on which values are permitted for a particular business case must be asserted. In HL7 this is known as Terminology Binding[5] and it asserts a specific relationship between a data element and certain Value Sets in the context of a particular business case, which is usually defined in an Implementation Guide. In implementation guides and some other standards artifacts', attributes and elements may be bound to Value Sets. Value Set bindings can be **STATIC**, meaning they are bound to a specified version of a Value Set and its specified codes or **DYNAMIC**, meaning they are bound to whatever the most current version of the Value Set definition is and therefore the codes specified to be included when the Value Set Expansion Code Set is created.

In cases of Value Set ***MODEL BINDING*** where the intended result is to allow use of any future Value Set Expansion *independent* of any specific Value Set Definition (i.e., **DYNAMIC MODEL BINDING** where use of the Value Set is not restricted to a single specific Value Set Expansion Code Set) then the Value Set binding should only include a Value Set Identifier and no static date.

[5]In the HL7 Core Principles Standard, the structure that implements terminology binding is the Value Set Assertion in section 5.2.2. The ISO standard 17583 Health informatics: Terminology constraints for coded data elements expressed in ISO harmonized data types used in healthcare information interchange describes the algorithms and structures using Value Sets for terminology binding in detail. Note this standard is in ballot as of July 2014.

In cases of Value Set model binding where the intended result is to bind to a specific Value Set Definition Version, *but* allow use of any updated Value Set Expansion based on that version, then a *version of the Value Set* bound should be identified with the specific Value Set Definition Version Identifier of interest. To be clear, because a single Value Set Definition Version can result in multiple Value Set Expansion Code Sets when the Value Set Definition Version is not LOCKED to a single Code System Version, including only a Value Set Version Identifier in (dynamic) model binding ***does not guarantee*** that a specific Value Set Expansion Code Set will always be used. Please see section 6.2.1 for more detail on the effect LockedDate has on a binding.

If the Value Set binding is to be a **STATIC MODEL BINDING** (the Value Set Assertion includes a *static date*), then it would be best, although it is not required, that the date populating *static date* be the same value as is used in the Value Set Expansion Date for the Value Set Expansion file containing the desired Expansion Code Set.

4.4 Value Set Definition Consistency in HL7 Models

HL7 models, in fact any model that must reference allowed sets of coded content, must reference these sets in some way. HL7 identifies such sets as "Value Sets" and the way this is done for HL7 v3 models is specifically described in the HL7 Core Principles section 5.1.3[6]. Value Sets are used throughout HL7 models including Version 2 models, although in Version 2 artifacts they have to date (this is under review) been represented in V2 tables which are not necessarily Value Sets or Code Systems. The newest HL7 model product, FHIR, also utilizes Value Set constructs[7], as do CDA-based artifacts. All of these models align, but details on the specific requirements necessary to describe and fully represent a Value Set have not been clearly documented. This document provides that detail and, based on participation of the membership involved, each of these HL7 products is expected to (eventually) be compliant with the standard. In that way Value Set content created in one product line for a particular business purpose could be usable in other models for the same purpose. There is no expectation that any particular Value Set must be re-used, but reuse is encouraged where appropriate.

In order to be certain that an implementation conforms to the rules in the Standards and the Implementation Guide, both structure and content must be taken into account. Both the specification of what it means to be conformant and the ability to construct testing tools

[6] http://www.hl7.org/documentcenter/public_temp_C01E0284-1C23-BA17-0C7D648F9680FEF0/standards/V3/core_principles/infrastructure/coreprinciples/v3modelcoreprinciples.html

[7] http://www.hl7.org/implement/standards/fhir/valueset.html

and procedures for content has been challenging. Formal and precise specification of the content and format of Value Sets as defined in this Standard for Trial Use (STU) provide a means to enable formal and strict (if desired) conformance specification and testing of content and support the ability to track conformance across version evolution.

4.5 Value Set Authoring and Maintenance

Value Sets represent the vocabulary constraint for a data element in a model for a particular use case and, thus, strongly influence the semantics that may be carried in the model. As such, they are usually authored and published using well-controlled processes to ensure quality and consistency. In order for such quality oversight and robust governance operations to be implemented efficiently, a well-designed standard for the content and structure for Value Sets is highly desirable. This Standard supplies such a definition and many of the elements described in the next sections are specifically included to support robust governance processes.

5 Value Set Definition Specification

5.1 Value Set Architecture

5.1.1 General Overview

When most people think of a Value Set, they think of a list of codes or phrases. These lists may be called member list, code list, etc. While a critical part of the usefulness of a Value Set, the final set of Code System members used by implementers, which is called the Value Set Expansion File in this specification, is only one part of the collective information that *is a Value Set*.

A Value Set is a specification composed of general metadata that is true for all versions of the Value Set[8] and version-specific information that describes the content of the Value Set[9]. Some of the information conveyed in the specification is not intended to be machine computable, but carries information users must interpret. Other elements in the specification are intended to be directly computable so that automated systems may act upon this information and generate reliable output.

Also included are critical metadata used to identify the people and entities that are responsible for developing and maintaining the Value Set Definition, Value Set Definition

[8]Green items in the Class Model Diagram.
[9]Yellow items in the Class Model Diagram.

versioning information and elements that can support tracking of changes to the Value Set and authoring workflow.

Each of the elements of the model is described in the sections below.

5.1.2 Relationship Between the Value Set Definition and Value Set Expansion Code Set

The phrase "Value Set" usually is taken to mean both of the following:

- ***Value Set Definition***, a description of the set of Concept Representations (usually codes) that are intended for use, **plus** the
- ***Value Set Expansion Code Set***, the set of resulting codes actually obtained for a particular use, drawn from one or more specific Code System versions.

Yet as is shown in this specification and in Core Principles, these are not exactly the same thing. In general, when the phrase "Value Set" is used (including within this specification), the intent is to reference both the Value Set Definition and the Value Set Expansion as one. From this point forward one of the more specific phrases is used when one or the other item is under discussion in particular.

A Value Set Definition is a set of metadata that describes the scope of the intended member Concept Representations, provenance of the included information and, most importantly, a set of instructions that describe (preferably in a computable manner) which code system Concept Representations should be in the Value Set Expansion Code Set. As such, the Value Set Definition is then applied to one or more Code System instances to determine the Value Set Expansion Code Set Member Concept Representations. In essence the Value Set Definition (specifically the Content Logical Definition described below) is a query into the Code System to retrieve the concepts as described in the definition. This is true *for every Value Set Definition*. It is not restricted to only definitions that use a logical or "intentional" definition; it is also true for simple explicit lists of individually selected concepts. As long as the Value Set Definition is not locked to a single Code System version, even simple code lists can result in changing Value Set Expansion Code Sets if the codes in the original definition are retired in later Code System versions.

Therefore a Value Set Definition *is not* the artifact that contains actual instance Value Set content. Only a Value Set Expansion (see Figure 1 and also Section 8 below) will contain the actual Expansion Code Set member Concept Representations that are the result of the proper Value Set Definition version applied against the proper Code System version.

5.1.3 Alignment with Intentional and Extensional

"Value Set Definition Methods" of HL7 Core Principles[10] describes two approaches to defining Value Sets, Intentional and Extensional, repeated in part here for clarity:

- *Extensional Definition*: Explicitly enumerating each of the Value Set concepts.
- *Intentional Definition*: Defining an algorithm that, when executed by a machine (or interpreted by a human being), yields such a set of elements.

Both approaches are used frequently in defining Value Sets but intentional definitions have sometimes been described in textual means that still require direct manipulation in order to determine a reliable set of concepts that make up the actual Value Set.

The Content Logical Definition section of this document describes how to represent a computable description of the Concept Representations that are intended to be in the Value Set Expansion. It is through the use of functions included in the Content Logical Definition that *any and all* concepts included in a Value Set Expansion Code Set will be identified. As such, the difference between Intentional and Extensional becomes essentially a description of the style used to determine the Value Set Expansion Code Set with one important distinction when considering Value Set maintenance and the Value Set Expansions that will occur with subsequent allowed Code System versions.

In an Extensional style Value Set if the Value Set Expansion is allowed to change with new Code System versions (i.e., is not LOCKED to a single Code System version), then new concepts will not automatically be added to the Value Set Expansion Code Set because each concept is explicitly identified, but concepts *may* fall out of the Value Set Expansion Code Set if they are retired. In an Intentional style Value Set the Content Logical Definition contains a logical expression that is to be evaluated against each allowed Code System version and that evaluation can result in both the addition and removal of concepts from the Value Set Expansion Code Set.

[10] 5.1.3.1 Value Set Definition Methods

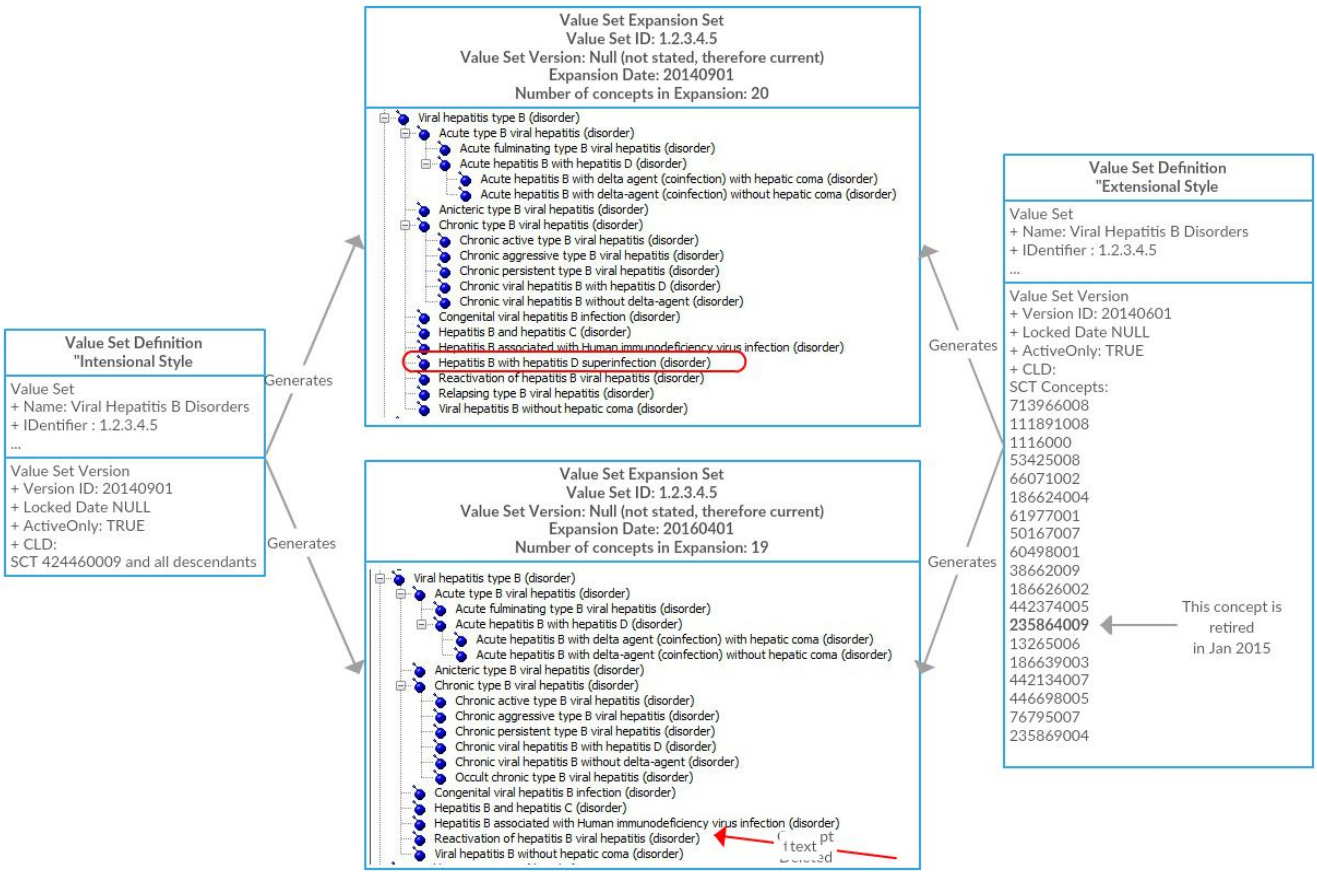

Figure 1 Value Set Definition types and resulting Expansions

5.1.4 Class Model Diagram

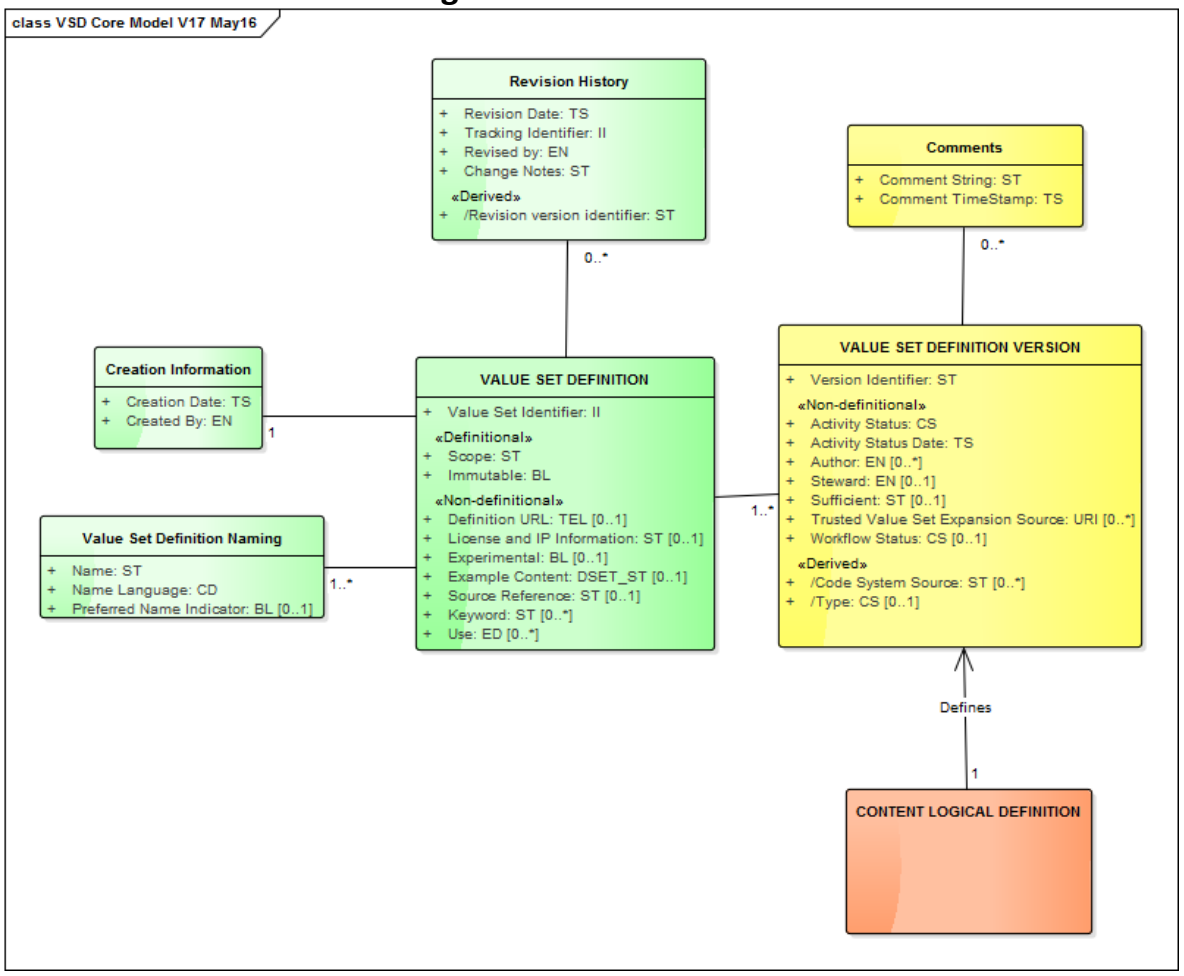

Figure 2 - Value Set Metadata

5.2 Elements

The Value Set Definition consists of a collection of Elements, which contain the specification and documentation of the Value Set. The Value Set Expansion can be generated from these elements and the documentation elements support governance and distribution. Any element with an unstated cardinality is presumed to have a cardinality of 0..*. All elements with a non-zero lower cardinality are required.

Each of the following sections is structured to align with the UML Model categories noted in the diagram above. Each of the primary sections below are sub-divided to indicate if the element described is definitional or non-definitional, where definitional elements, when changed, *must* result in either an entirely new Value Set (for the value set elements) or a new value set version (for value set version elements). Elements that are derived from other elements are also identified. Each section below describes the details of the components illustrated in Figure 2 - Value Set Metadata.

5.2.1 Implied Constraints of "Definitional" and "Non-definitional" Elements

The class diagram includes "definitional" and "non-definitional" categorization of model elements. By this we mean that when elements in a class that are "definitional" are changed in a substantive way, the model expects that a new instance of the class should be created. For all elements, except those that are a ST datatype, any change is potentially substantive. For the Scope content, which is a text string, a substantive change is subjective and described more completely in the Scope section. The definitional elements impose baseline process expectations in the standard, i.e., all compliant implementations of the standard must create new Value Set Definitions or new Value Set Definition versions when a definitional element has a substantive change. A new Value Set Definition or Value Set Definition version that is based on a change in a non-definitional element is not considered best practice, but may occur in certain business scenarios. When a new Value Set Definition Version or Value Set Definition is created under such a circumstance, this does not change the definitional criteria. For example, in the VSAC, "Steward" is a Value Set Definition version element that has been designated as *definitional* so when changed, will always result in a new version of the Value Set Definition, even if no other items have been changed.

5.2.2 Value Set Definition

Elements described in this section provide identification and descriptive information about the entire Value Set Definition and can be consistently applied to all versions of the Value Set Definition. This collection of elements is made up of an identifier and two categories of model elements: definitional and non-definitional. These categories are described in Implied Constraints above.

5.2.2.1 Value Set Identifier

Definition: globally unique string that characterizes this Value Set Definition
Description: The string must be globally unique and a specific Value Set Definition can be referenced in other artifacts using this string.
Usage Notes: Current implementations at HL7 use OIDs or, more generally, URIs. GUIDs or RUIDs are also used. UID Data type specifies each allowed sub-type (OID, RUID, UUID, URI) which is self-described in the identifier instance. A Value Set Definition can only have a single identifier of any sub-type and each identifier MUST identify the same specific Value Set Definition. Conformant cardinality is 4 because there are 4 allowed sub-types that can be used and there can only be a single canonical persistent identifier for each sub-type. If an implementation allows multiple identifiers that can be evaluated to obtain the single Value Set Definition, one identifier is to be selected as the persistent single identifier.
Cardinality: 1..4
Data Type: UID

5.2.2.2 Value Set — Definitional elements

If either of the elements within this section changes substantively, then a new Value Set Definition **must** be created and a new Value Set Identifier assigned.

5.2.2.2.1 Scope

Definition: textual description of the span of meanings for concepts to be included within the Value Set Expansion Code Set including the intended use and limitations of the Value Set.
Description: This is intended to convey between humans (and not necessarily in a computable fashion) the scope of Concept Representation meanings and capture important elements of the context of use for the Value Set. This should describe "the semantic space" to be included in the Value Set Expansion. This can also describe the approach taken to build the Value Set Expansion Code Set. Any ***substantive change*** is one that changes the meaning of the scope and therefore implies a different Value Set Definition rather than a new Value Set Definition Version. This item is a key component for the meaning of the Value Set.
Usage Notes: The Scope should cover the following kinds of information:

- Focus: The general focus of the Value Set as it relates to the intended semantic space. This can be the information about clinical relevancy or the statement about the general focus of the Value Set, such as a description of types of messages, payment options, geographic locations, etc.
- Model Context: A statement that describes how this Value Set is to be used in an artifact.
- Inclusion Criteria: Criteria describing what concepts or codes should be included and why.
- Exclusion Criteria: Criteria describing what concepts or codes should be excluded and why.

As noted above, the scope text is intended to be non-computable, therefore determining a *substantive change* is solely the responsibility of the Value Set steward/author/reviewer community. The expectation is that the **original intent** of the Value Set should be held unalterable.
Data Type: ST
Cardinality: 1..1

5.2.2.2.2 Immutable

Definition: property indicating that no change to the Content Logical Definition may occur. Immutable means no change can occur.
Description: If this is set to 'true', then *no new versions* of the Content Logical Definition can be created. Other metadata might still change and it is possible that the Value Set Expansion Code Set can change if the underlying Code Systems change and the Content Logical Definition (CLD) is not locked to a specific Code System version.

Usage Notes: Note that the implication is that if this is set to 'true', there may be only one Value Set Version for this Definition. The default value is FALSE. The Immutable Flag tends to be set to 'TRUE' in one of two cases:

- Where the Value Set Definition, by the nature of its usage, cannot change (e.g., "all specializations of ACT in ActClassCode")
- Where there's no safe way to express the "Scope" such that someone else could safely make changes to the Value Set Definition.

Source workflow control must guarantee that the same URI always yields the same definition.
Data Type: BL
Cardinality: 1..1

5.2.2.3 Value Set Definition— Non-definitional Elements

The content of the elements in this section may change without requiring a new Value Set Definition be created.

5.2.2.3.1 Value Set Definition Naming

Definition: contains the human-readable monikers of this Value Set Definition.
Description: This repeating group of elements contains information about the human-readable strings associated with this Value Set Definition that are used as names.
Usage Notes:
Data Type: Group of elements (listed following)
Cardinality: 1..*

5.2.2.3.1.1 Name

Definition: word or set of words by which the Value Set is known, addressed or referrenced
Description: This name is intended to be human readable, short and as specific as possible and it should relate to the content and intent. It is considered to be the name of the Value Set Definition.
Usage Notes: This need not be unique. Some use cases may require uniqueness within a namespace and, therefore, best practice is to make the name unique.
Data Type: ST
Cardinality: 1..1

5.2.2.3.1.2 Name Language

Definition: code to indicate the system of communication used by a particular country or community in which the name is represented.
Description: This indicates the language system (e.g., US English) in which the name is expressed.
Usage Notes: The intention is to use the IETF language tags preference referenced by the current RFC for BCP 47.

Data Type: ST (drawn from IETF language tags noted)
Cardinality: 1..1

5.2.2.3.1.3 Preferred Name Indicator

Definition: identifies the single preferred name for the "Name Language".
Description: Flag that this *Name* in this *Name Language* is the preferred human-readable signifier in this language. The only permissible value of this property is "preferred".
Usage Notes: There may be multiple human readable names in a given language, and this flag indicates which of them is preferred for the given language. The expectation is that only one is preferred for any given language.
Data Type: ST
Cardinality: 0..1 (within each Value Set Naming group of elements)

5.2.2.3.2 Definition URL

Definition: a web pointer to where the Value Set Definition may be located.
Description: Pointer to the authoritative accessible, persisted source of truth of the entire Value Set Definition, including textual information and available versions.
Usage Notes:
Data Type: URL
Cardinality: 0..1

5.2.2.3.3 License and IP Information

Definition: publishing restrictions for the Value Set Definition metadata
Description: These are generally legal restrictions on the use and publishing of the Value Set Expansion and metadata.
Usage Notes: This is intended to convey the restrictions of the directly referenced Value Set Definition and Expansion, including Code System(s). It is a text block of unlimited length and unspecified internal format.
Data Type: ST
Cardinality: 0..1

5.2.2.3.4 Experimental

Definition: this Value Set is created for educational or testing purposes, and is not intended for production use.
Description: When set to 'experimental', this Value Set should not be used in a production system or environment and may be used for exemplary purposes. It must be noted that use of a Value Set Definition designated as Experimental is under the control of the user and is not be controlled by this element. If not populated, the Value Set may be used in a production system or environment. This decision is wholly the judgment of the Value Set steward.
Usage Notes: When absent, the Value Set may be used for any purpose. Value Sets noted to be Experimental may also be used for demonstration purposes.

Data Type: ST
Cardinality: 0..1

5.2.2.3.5 Example Content

Definition: set of easily understood concepts that will be found in the Value Set Expansion Code Set
Description: The collection of examples is an illustrative list of human readable values that are expected to be found in the Value Set Expansion of this Value Set Definition. It is useful both in the authoring process and as a convenience for reviewers, especially if the Value Set Expansion Code Set is large. It should contain a small (≥ 3) set of example values. This content must include at least one code, description, or definition for each example.
Usage Notes: It is most useful if the examples included are human-readable, therefore if the code is not human readable, the description and/or the definition should also be included.
Data Type: ST*
Cardinality: 0..1

5.2.2.3.6 Source Reference

Definition: reference to prior work used as a basis in authoring of this Value Set
Description: This text is intended to act as a citation to work done elsewhere that is not part of the current stewarding process where the referenced source is in some way a basis of the current Value Set Definition.
Usage Notes: This may refer to other Value Set Definitions, research articles, or other resources. This is not intended to document uses of the Value Set Definition, for this, see "Utilization" section below. For example, the identifier for a Value Set Definition that was cloned to begin the work on this Value Set Definition, or a pointer to a published article that describes appropriate concepts.
Data Type: ST
Cardinality: 0..1

5.2.2.3.7 Keyword

Definition: a word or phrase that captures a key aspect of a Value Set Definition
Description: Word or words that may be used in an information retrieval system to index the contents of the Value Set Definition. The string datatype allows implementers to use strings or coded concepts.
Usage Notes: This may be used to surface indexing terms for a search.
Data Type: ST
Cardinality: 0..*

5.2.2.3.8 Use

Definition: the manner in which the Value Set Definition will be employed or applied.

Description: an optional repeating element used to capture information about consumers of the Value Set Definition and the implementations, projects or standards where the author has utilized the Value Set.
Usage Notes: When initially conceived (and in the initial publication) this was a linked set of entries to describe a User who is employing the Value Set for a particular Usage. In the comment review process a linked set of "User" and "Usage" was deemed too complex, so this element is now a general data type. Users may implement anything from a simple blob of text to something more complex. The information captured in "Use" can and should include names of programs and/or names of organizational entities that use the value set definitions, including standards that require the value set definition. This is likely to be a "point in time" view and should not be considered an authoritative listing of all uses of the Value Set. In the USA, the Value Set Expansions created for electronic quality measures (eCQMs) that are a part of the Meaningful Use requirements could implement this as follows:

- User: Measure Steward name; Usage: Measure Identifier & version/Name
- User: Centers for Medicare & Medicaid Services (CMS); Usage: eCQM release identifier.

Data Type: ST
Cardinality: 0..*

5.2.3 Value Set Definition Version

The Value Set Definition Version section contains the information required to computationally produce a Value Set Expansion from the Value Set Definition, metadata specific to the version and an identifier for the Value Set Definition Version. Similar to the overall Value Set Definition, the Value Set Definition version metadata includes an identifier, elements characterized as definitional, elements characterized as non-definitional (both are described in Implied Constraints) and a derived element. Each of the elements in this section applies to a particular version of a Value Set Definition.

5.2.3.1 Value Set Definition Version Identifier

Definition: a unique string labeling a Value Set Definition Version.
Description: A unique identifier within the context of the Value Set Definition that is used to indicate a specific Value Set Definition at a point in time.
Usage Notes: The identifier must be updated when any of the "Definitional metadata elements" of the Value Set Definition have changed. In addition, it may be updated under other circumstances. This may be a string identifier of any type or may explicitly be a timestamp, which is often used to simplify dynamic binding processing. If the string is intended to be encoded as a timestamp, the format should be TS.DATETIME.FULL to be consistent with HL7 Datatypes R2.
Data Type: ST
Cardinality: 1..1

5.2.3.2 Value Set Definition Version — Definitional Elements

If either of the elements within this section changes in any way[11], then a new Value Set Definition Version **must** be created with a new Value Set Version Identifier assigned. The Value Set Identifier *does **not** change*.

5.2.3.2.1 Content Logical Definition

Definition: formal representation used to determine the coded concept contents to be included in the Value Set Expansion Code Set
Description: The Content Logical Definition (CLD) is a formal representation of Value Set content and is intended to be machine processable. It is used to generate the coded content of the Value Set Expansion, i.e., the Value Set Expansion Code Set.
Usage Notes: This element uses the ED data type[12], meaning that the content is structured and that the structure includes an identification of the structure (or the application that must be used to interpret the data). For example, an "HL7 Expression" uses a MIME type of "application/xml". This flexibility means that subsequent specifications can identify the allowance for additional expression syntaxes.
The HL7 Expression functions are an available syntax for the CLD that is described below, possibly including some recursive elements. Each recursive section is considered a "clause". The Value Set Definition Version contains a single mandatory Content Logical Definition that describes different types of specifications of content, some of which contain or reference Content Logical Definitions; thus the definition is inherently recursive. The list of functions is described in Section 6.
Data Type: A group of elements (see Content Logical Definition).
Cardinality: 1..1

5.2.3.3 Value Set Definition Version — Non-definitional Elements

The elements within this section *may change* without requiring that a new Value Set Definition Version be created. Some elements in this section should never result in a new version (e.g., Activity Status, Effective Date, Workflow Status, Completeness), but a change in others could lead a Value Set steward to decide to identify a new version (e.g., Steward, Author, etc..).

---

[11] If a content logical definition content expression contains Concept Representations that can change yet are still considered a representation of the same concept, then the value set version may change due to a Code System Version change alone.

[12] For more specific information on ED, see https://www.hl7.org/documentcenter/public_temp_8AAA341E-1C23-BA17-0CB0D645248F7413/wg/inm/datatypes-its-xml20050714.htm#dtimpl-ED

5.2.3.3.1 Activity Status

Definition: the release and/or usability state
Description: The state that represents the current stage of use of the Value Set Definition.
Usage Notes: This element has the following allowed values:

- Preliminary: State during which time the Value Set Definition version is being drafted and is not available for use. The element Workflow Status will carry additional information regarding pre-Active state.
- Active: State during which the Value Set Definition version is available for use. The Activity Status Date associated with this status is known as the "Effective Date".
- Inactive: State indicating that the Value Set Definition version is no longer available for use in creating new content. The Activity Status Date associated with this status is known as the "Expiration Date".
- Deleted: State intended to be used to remove a Value Set Definition from use and view in a repository and is only possible if the Value Set was never Active (i.e., can only transition from Preliminary).

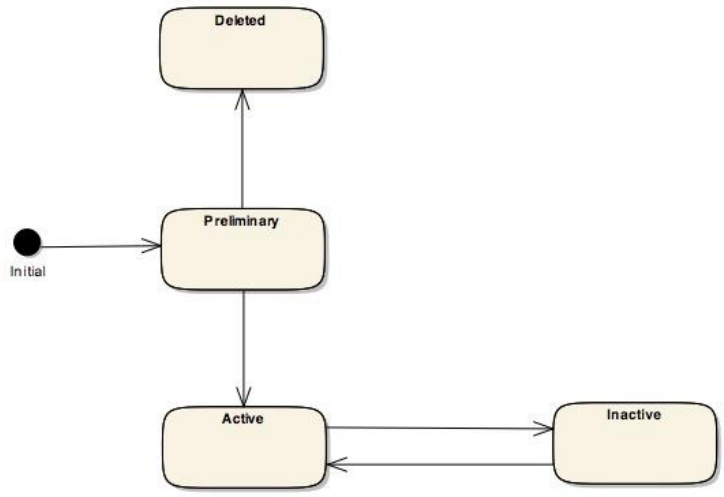

Figure 3 - Valid state transitions for Value Set Activity Status

Since activity status is only represented here in Value Set Definition Version information and is a required element, then any Value Set Definition that does not have this is *not valid.* Furthermore, a Value Set Definition may have more than one Value Set Definition version with any of the status states noted. Given that this can lead to overlapping Active definitions, external governance must be available to either restrict the allowed overlap or resolve overlap if usage does not specify a particular Value Set Definition Version.
Data Type: CS
Cardinality: 1..1

5.2.3.3.2 Activity Status Date

Definition: the date when the associated Value Set Definition Version activity status is set.
Description: This is the date the associated status for this version is to begin.

Usage Notes: When the Activity Status is set to "Active", the Activity Status Date defines the Effective Date which is the first date-time the Value Set Definition Version becomes active. When the Activity Status is set to "Inactive", the Activity Status Date is the first date-time when the Value Set Definition version becomes Inactive. The start Date_time is expected to be as of 0001 UTC of the Activity Status Date.

It is *strongly encouraged* that the Activity Status be set such that **no more than one Value Set Definition Version for a single value set ID can have an Activity status of ACTIVE at the same time within a single realm**. In cases where this is not true, evaluation of the alignment of a Value Set Expansion Code Set to a specific Value Set Definition, as referenced in a CD, will be problematic.
Data Type: TS
Cardinality: 1..1

5.2.3.3.3 Workflow Status

Definition: the state of development of the Value Set while in a single Activity Status.
Description: Workflow Status is used to represent details of the Value Set Definition development process. The development of a Value Set Definition often follows a formal workflow process from initiation to completion and this element carries the state variable for this state machine. The assumption is that when first created, a Value Set Definition would have a workflow state of Draft. Additional workflow states may be used.
Usage Notes: The values that are traditionally used for this element while the Value Set Definition has an Activity Status of Preliminary are assumed to include phrases that capture various stages in review and approval. The US Value Set Authority Center (VSAC) uses the following workflow statuses. The Activity Status is "Preliminary" (see Activity Status) for all these workflow statuses:

- "Draft" – The initial Value Set Definition creation status. An author can make changes to the Value Set Definition only while the definition is in this status.
- "Proposed" – Once an author has completed editing, the Value Set Definition is submitted for review by a value set steward. The submission changes the value set definition to this status and editing can no longer occur.
- "Approved" – When the Steward completes the review and approves the Value Set Definition. This status is for Value Set Definitions that have successfully completed a review process, *but are not yet set to be published*.
- "Ready to Publish" – This status occurs when the steward sets a Publication Date that defines when the Value Set Definition version is to become ACTIVE (Activity Status = ACTIVE). The Value Set Definition version is in this status from the time the publication date is set until the publication time. VSAC Value Set Definition versions are published at 00:01 (one minute after midnight) of the publish date morning (Eastern Time) and the earliest a Value Set Definition version can be published is the next day's morning.

VSAC allows all Value Set Definition workflow statuses to be changed back to the prior status as long as the Value Set Definition has not been "Published" (Activity Status = ACTIVE). A Value Set Definition version can only have one workflow status at any time.

There may be additional states defined by different developers. This is an optional element because the use of Activity Status “Preliminary” may be sufficient for some implementations.

Data Type: ST
Cardinality: 0..1

5.2.3.3.4 Steward

Definition: the entity that is responsible for the content.
Description: This is a textual description of the organizational entity responsible for the content and maintenance.
Usage Notes: The information included should include contact information.
Data Type: ST
Cardinality: 1..1

5.2.3.3.5 Author

Definition: the entity or set of entities that create and may modify the Value Set Definition content.
Description: The name of a group or an individual.
Usage Notes: This can be any combination of groups or individuals. When known and actively maintained, this should be populated. The information included about the Author may include contact information.
Data Type: ST
Cardinality: 0..*

5.2.3.3.6 Comments

Definition: human-specified notes and other documentation
Description: This optional repeating group of elements contains human-specified notes and other documentation, which consists of a time-stamped list of comments text blocks.
Usage Notes:
Data Type: Group of elements (listed following)
Cardinality: 0..*

5.2.3.3.6.1 CommentString

Definition: unrestricted field of remarks or other text.
Description: Each comment is a time-stamped entry of arbitrary length that is only editable by those in the author group.
Usage Notes:
Data Type: ST
Cardinality: 1..1

5.2.3.3.6.2 CommentTimeStamp

Definition: date/time when the comment was created
Description: The time stamp for the comment
Usage Notes:
Data Type: TS
Cardinality: 1..1

5.2.3.3.7 Sufficient

Definition: indicates that a Value Set Definition generates an Expansion Code Set that fully implements the Value Set Definition scope.
Description: A Sufficient Value Set Definition provides all the concepts indicated by the scope. If a Value Set Definition is "not sufficient", then there may be a need for additional local codes or future changes to the Content Expression. Only one value is allowed if the Value Set Definition is "Sufficient". If Null, then the Value Set Definition is "not sufficient".
Usage Notes: This is intended to allow Value Set Definition authors to convey expected binding semantics to users of the Value Set Definition. If the Sufficient attribute is "Sufficient", then the Value Set should be bound with the expectation that no further content is currently expected. In HL7 v3 semantics this would be Coded No Extensions (CNE).
Data Type: ST
Cardinality: 0..1

5.2.3.3.8 Trusted Value Set Expansion File Source

Definition: list of references to trusted Value Set Expansion files.
Description: This is to be used to provide links to services for existing persisted Value Set Expansion files that are known by the Value Set author where the Value Set Expansion file content can be trusted.
Usage Notes: This URI should directly yield the Value Set Expansion file.
Data Type: URI
Cardinality: 0..*

5.2.3.4 Value Set Definition — Derived Elements

Elements in this section are not entered by the author, but are derived values based on other elements in the Value Set Definition metadata.

5.2.3.4.1 Code System Source

Definition: the code system(s) identifiers explicitly referenced by the Value Set Definition's Content Logical Definition.
Description: An exhaustive list of all Code Systems referenced in the Content Logical Definition.

Usage Notes: This list of Code System Identifiers should be derived from the Content Logical Definition to ensure that it is accurate. When the Content Logical Definition does not specify any content types based on Code Systems, this is not populated. It is possible that a Code System will be in the CLD *but not generate a concept in the Value Set Expansion Code Set;* therefore, this may list Code Systems that are not resident in a specific Value Set Expansion Code Set.
Data Type: UID (OID, UUID, URI or RUID)
Cardinality: 0..*

5.2.3.4.2 Type

Definition: indicator of Value Set Definition being Extensional (Enumerated) or Intentional (Criteria-based) or Grouping (only references other Value Sets).
Description: This is derived from the Content Logical Definition and is not a unique definitional element.
Usage Notes: The following strings are intended to be a characterization of the Content Logical Definition:

- Extensional: the CLD is a simple list of individually specified concepts that uses no relationships or other attributes to determine the Value Set Expansion Code Set.
- Grouping: the CLD is restricted to a UNION of one or more other Value Set references.
- Intentional: Any CLD that is not either Extensional or Grouping.

Data Type: CS ("Intentional", "Extensional", "Grouping")
Cardinality: 0..1

5.2.4 Creation Information

Creation Information contains the initial creation user data for the Value Set Definition.

5.2.4.1 Creation Date

Definition: when the Value Set Definition was created.
Description: This element records the date-time when the first draft of the Value Set Definition was saved.
Usage Notes: The expectation is that if the creation date is unknown, that one will be asserted when the Value Set Definition is entered into the system.
Data Type: TS
Cardinality: 1..1

5.2.4.2 Created_by

Definition: name of the user entity that created this Value Set Definition.
Description: This records the entity that performed the original Value Set Definition creation step. This data will persist for the life of the Value Set Definition (and is consistently represented in all versions) because the original creation occurs only once.

Usage Notes: In VSAC, this is the logged in user that saved the first view of the Value Set Definition. In most instances it would be best if this were the original Value Set Definition steward to capture that information without requiring a query into the Value Set Definition modification history.
Data Type: ST
Cardinality: 0..1

5.2.5 Revision History

Revision History is used to record individual changes to any element of the Value Set metadata. It is intended to capture changes over time and serves as an audit trail of all such changes. Any change of any kind can result in a revision history item.

5.2.5.1 Revision Date

Definition: date and time when an element was changed in some way.
Description: The Date_time when a Value Set Definition element was stored
Usage Notes: This should be captured down to the second.
Data Type: TS
Cardinality: 1..1

5.2.5.2 Tracking Identifier

Definition: identifier for the changes for each revision
Description: Unique identifier for the change captured in the revision history.
Usage Notes: This may be used to disambiguate multiple changes and revisions that all carry the same timestamp, such as those done for publishing reasons. It may be best if this is a GUID.
Data Type: ST
Cardinality: 0..1

5.2.5.3 Revised By

Definition: name of user entity responsible for the revision.
Description: Carries the name or identification of the user entity that is considered to have made the revision.
Usage Notes: Some systems will assign this as the entity that was logged in when the revision occurred.
Data Type: ST
Cardinality: 0..1

5.2.5.4 Change Notes

Definition: annotation of revision

Description: Specific narrative description and/or explanation of the changes made to the Value Set Definition, e.g., what was done and perhaps why.
Usage Notes:
Data Type: ST
Cardinality: 0..1

5.2.5.5 Revision History — Derived Element

5.2.5.5.1 Revision Version Identifier

Definition: the Value Set Definition Version Identifier to which the history item applies.
Description: A derived element that captures the current Value Set Definition Version Identifier at the time the revision history item was created.
Usage Notes:
Data Type: ST
Cardinality: 1..1

6 Content Logical Definition

The formal specification in the Value Set Definition of the Code System content that is to appear in the Value Set Expansion Code Set is called a Content Logical Definition (CLD.) In the initial publication for review of this specification, only one formalism for representing the CLD was described. Based on considerable feedback, this final publication of the specification allows the CLD to be specified in any usable way, including the original grammar now called HL7 Expression grammar. While this change increases the possibility of more than one standards-compliant way of representing a CLD, it provides flexibility to support existing formalisms for obtaining concepts/codes from code systems. This may mean that a specific Value Set Definition will use a CLD formalism that all users may not completely understand, but the intent is to expect designers of such formalisms, such as the IHTSDO, to publish descriptions of the formalism used so others may understand and compute the Expansion Code Sets defined. At a minimum, the HL7 Expression grammar serves as a "default" approach through which the key functions are described. This updated specification also separates out the "Non-computable Expression" type of CLD so that an expression that describes, but does not support formal computation of the Value Set Expansion Code Set is available for those that need it. This is meant to be easily identifiable as "non-computable" and, as such, a Value Set Definition that uses this in a CLD cannot mix together into one Value Set Definition version both non-computable and formal syntax elements.

The CLD consists of a single sub-type of Content Defining Element, which by nature may be recursive. Further, most sub-types of Content Defining Element refer to a specific Code System and to a set of constraint parameters relative to that code system. In aggregate

these comprise an expression that precisely specifies coded content to be generated into a Value Set Expansion Code Set.

6.1 Content Logical Definition General Model

A Value Set Definition has a Content Logical Definition. The Content Logical Definition describes how the Value Set Expansion Code Set shall be generated. The UML diagram in Figure 4 (below), illustrates that a CLD is a content expression that can use only a single syntax. This document describes a default expression syntax called "HL7 Expression."

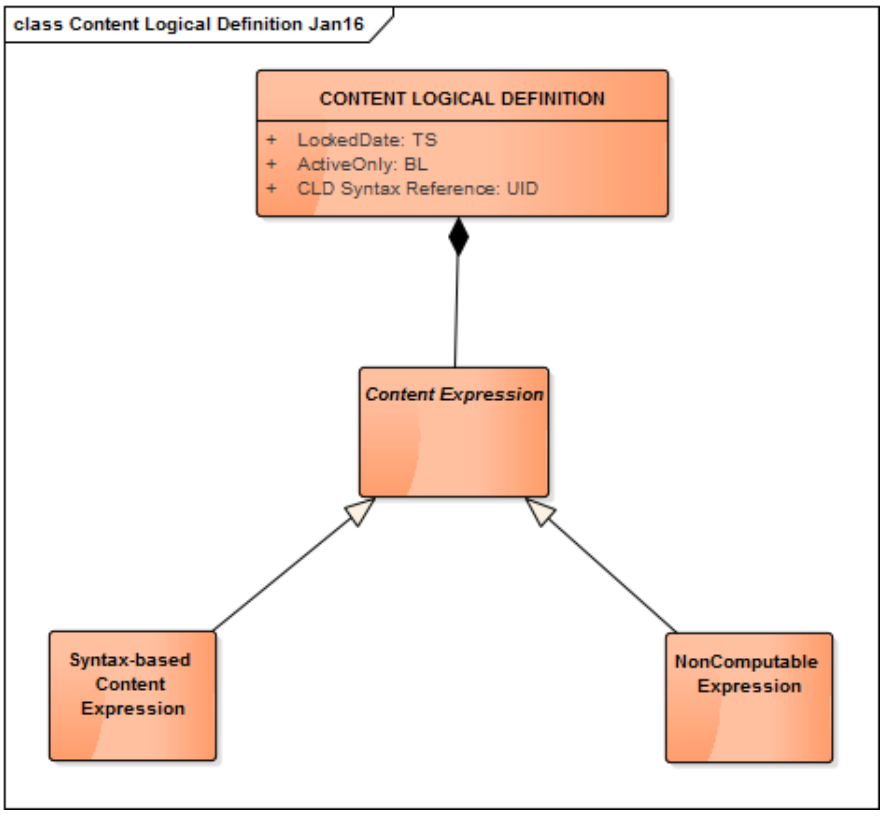

Figure 4 - A **Content Logical Definition** is made up of a Content Expression (that can recursively contain other Content Expressions using the same grammar). The Content Expression can be an "HL7 Expression" using the functions described in this specification, it can be some other syntax or it can be a textual description of the process to be followed to obtain the correct concepts and, as such, is considered "non-computable".

As is noted in the figure above and described in the caption, a CLD is expected to contain a formal specification of the set of Concept Representations to be retrieved from one or more code systems. A "syntax-based" formalism is expected to be computable, wherein a "Non-

computable" expression is assumed to be a textual description and not directly computable. The default syntax for the CLD is HL7 Expression grammar. This set of expression functions is based on the HL7 Model Interchange Format[13] that is described in detail below as an illustrative enumeration of useful functions for obtaining concepts from a code system. The three types of expressions noted in the figure are described below.

6.2 Content Logical Definition – Elements

6.2.1 LockedDate

Definition: the effective date that is used to determine the version of all referenced Code Systems and Value Set Definitions included in the Content Expression that are not already tied to a specific version.
Description: If a LockedDate is present, this Value Set Definition version can only generate a Value Set Expansion Code Set that is defined by evaluating the Content Logical Definition against the most recently available version of the Code System as of the LockedDate and the most recent Value Set Definition version as of that date for any Value Set Definitions included by reference. If no LockedDate is provided, then this version of the Value Set Definition could produce different Value Set Expansion Code Set content with every new Code System version used and Value Set Definition version referenced.

A LockedDate provides the author the ability to restrict the Value Set Expansion Code Set content independent of Value Set use (e.g., as defined as part of HL7 v3 Binding Stability) by fixing the Code System Versions needed to determine the Value Set Expansion Code Set to the most recent Code System Version as of the LockedDate (e.g., the Code System versions and Referenced Value Set Definition versions in the CLD).

When specified, "locking" is transitive to all referenced Value Set Definitions that are not already locked within the CLD, where in those cases the "inner" specified Code System version takes precedence (e.g., the code system Version used to determine the Value Set Expansion Code Set content of a CLD "clause" is governed by the Code System version specified "closest" to the clause.) By "transitive" we mean that "locking" of a clause within a CLD to a Code System Version takes precedence over "locking" performed by setting the value of the Value Set Definition LockedDate. This is true where the clause is directly part of the CLD but also if that clause is itself nested within a Value Set Definition of a Value Set Definition which is referenced in the CLD.

When a LockedDate is specified, the Value Set Definition Version is considered "Locked" for all uses independent of binding "stability", i.e., when ***no*** model binding stability date is specified, thus making the binding "Dynamic", the Value Set Expansion Code Set provided by a Value Set Definition with LockedDate specified ***will always be the same expansion,***

[13] http://www.hl7.org/implement/standards/product_brief.cfm?product_id=101

the one determined by the LockedDate. Otherwise, the Value Set Definition is considered to be "unlocked" and may have different Value Set Expansion Code Sets as underlying Code Systems and/or Value Set Definitions evolve.

Usage Notes: *A LockedDate is best used when there is a need to lock all referenced Value Set Definitions that are included in the CLD to a specific Code System version*. If an author wishes to lock a small subset of concepts needed for the Value Set Definition to a specific Code System version (versus referenced Value Set Definitions), the author should use the appropriate function in the CLD grammar chosen[14] and make the "lock" occur *within the CLD clause grammar* and not something that is applied across the entire CLD. The LockedDate is used as the effective date to determine the most recent versions of all referenced Code Systems and referenced Value Set Definition versions as of that specified date. If a Code System version is defined on the LockedDate, that new version is the "most current version".

Given that LockedDate affects the Value Set Expansion Code Set to be used in an implementation, independent of the Binding Stability within a model binding, and that LockedDate supersedes the "Static Date" used to define model binding stability, it is important for model binding to consider the CLD content. Since Value Set Definition LockedDate dictates the Value Set Expansion Code Set independent of use, Binding Stability will have *no impact* on the ability of a Value Set Definition version with a Locked Date defined to generate different Value Set Expansion Code Sets as the Code System version changes. i.e., Binding Stability is trumped by a Value Set version with LockedDate defined.

In addition, the Binding Stability date in a "static binding" is intended to *bind to the most recent Active Value Set Definition Version* as of that (stability) date and then apply the definition to the most recent Code System version as of the stability date. In all cases a LockedDate (or when using the HL7 Expression grammar, a defined VersionDate in a DrawnFromCodeSystem element) always takes precedence over a Binding Stability date.

Table 1: Binding + Stability Date Interaction Matrix

Model Binding Stability	Value Set Definition Version Lock State	
	Locked Date set	No locked date
Dynamic Binding	Locked Value Set Expansion Code Set version is all that is available	Value Set Expansion Code Set content may change with new Code System version (Dynamic changes occur)
Static Binding	Locked Value Set	Value Set Expansion Code

[14] Using the HL7 CLD grammar the appropriate function to use in a CLD clause is VersionDate within CodeSystemForExpansion.

	Expansion Code Set version is all that is available	Set content fixed to expansion current as of Static binding date

Data Type: TS
Cardinality: 0..1

6.2.2 ActiveOnly

Definition: include only ACTIVE Concept Representations in the Value Set Expansion Code Set
Description: This attribute flag indicates if "only active" Concept Representations should be included in the resulting Value Set Expansion Code Set when executing the CLD query. An active Code System Version concept is identified by the "activity status" which is meant to indicate if the concept may be used for data collection and recording at the time the Code System Version is the currently active Code System.**Usage Notes**: The default is "FALSE" which means that *all concepts* that match the CLD in the Code System version used to determine the Value Set Expansion Code Set will be returned. Any Code System concept attribute that represents activity status should be ignored if ActiveOnly = FALSE. If a Code System has no concept attribute that represents activity status, then all concepts should be considered to have an activity status = "ACTIVE". It is assumed that a concept activity status of "DEPRECATED" or "RETIRED" or "INACTIVE" all represent a status that is **NOT** ACTIVE. This attribute is to be implemented independent of "LockedDate" such that if LockedDate is set, then this attribute defines if only active concepts or all are placed into the Value Set Expansion Code Set.
Data Type: BL
Cardinality: 1..1

6.2.3 CLDSyntaxReference

Definition: a link or identifiable name for the syntax used in the Content Logical Definition Content Expression.
Description: This refers to the syntax used in the Content Expression. If this element is null, then the Content Expression is non-computable.
When populated, this element should provide a link that uniquely specifies the syntax used for the Content Logical Definition. When populated, at a minimum it should provide a unique name that can be used to find documentation of the syntax. This element serves as both a pointer to the documentation and an identifier of the syntax.
Usage Notes: A URL is preferred. It is suggested that this element may be changed ***without being considered a change in the CLD.*** This means this element may change without triggering the requirement for a Value Set Definition Version update.
The Content Expression syntax described in the An HL7 Value Set Definition Expression Syntax section will use a URL that points to the published version of this document.
Data Type: UID
Cardinality: 0..1

6.2.4 Content Expression

The Content Expression holds the text (potentially combined with the LockedDate) that is used to determine what Concept Representations (codes) should be included in a Value Set Expansion Code Set. There is great value in making this text directly computable, which is supported by CLDSyntaxReference that identifies the expression syntax used by the Content Expression text. Based on comments received on the initial version of this document, this updated specification is purposefully open regarding how a Content Expression can be represented. Even so, there are two approaches to the information provided in the Content Expression: those that are computable and therefore are based on the use of a specific evaluable syntax (and therefore should have a CLDSyntaxDescription) and those that are not using a computable syntax, are non-computable and essentially contain textual guidance for how the codes to be included in the Value Set Expansion Code Set are to be identified. In the sections below the types are discussed further.

Definition: a computable string that when evaluated using the appropriate syntax yields a set of Concept Representations from identified code systems.
Description: This string contains the entire expression rendered using the syntax noted in CLDSyntaxReference. When evaluated based on an approach appropriate for the defined syntax, this should result in identifying a Value Set Expansion Code Set of Concept Representations (usually codes) from the Code Systems identified within the expression.
Usage Notes: Given that the entire expression used to determine the Value Set Expansion Code Set that will be included in this element, the size should not be significantly restricted.
Data Type: ST
Cardinality: 1..1

6.2.4.1 Syntax-based Content Expressions

As has been described elsewhere in the document, the intention of this specification is to support the computable determination of specified Concept Representations (codes) from Code Systems as delivered in Value Set Expansion Code Sets for use in data capture and exchange. To support ease of creation, maintenance and use, it is preferred that the Content Expression is directly computable. To do so, the syntax used to represent the "expression" used to compute the Value Set Expansion Code Set must be clearly specified and allow full use of the Code System (and additional data sources) to determine what codes are wanted. A complete set of syntax functions that may be used to fully specify a Content Expression is described in An HL7 Value Set Definition Expression Syntax section. This section should be considered a default HL7 Content Expression Syntax.

In addition to the Default HL7 Expression Syntax just noted, other syntaxes may be used for a Content Expression. Support for "Other Syntax Expression" types means that any reasonable syntax can be used to computationally describe how to identify a specific set of Concept Representations (usually codes) from Code System(s). This is a change from the initial version of this specification where only what is now called "HL7 Expression" syntax was supported.

An incomplete list of Other Syntax examples that may be used includes:

- SNOMED CT Expression Constraint Language[15]
- OWL[16]
- SQL[17]
- Apelon Terminology Query Language (TQL)

6.2.4.2 Non-computable Content Expression

Some descriptions of Code System content cannot be captured in a computable syntax. This can occur for a variety of reasons, such as when the needed codes are not identified using computable parameters (e.g., "the codes representing conditions currently under discussion in the US Public Health blog") or when an initial version of the CLD is crafted as a general textual statement using "pseudo-code". In this case the CLD can use text to capture the CLD mechanics. As noted before, if a CLD uses this approach, it cannot be combined with a formal computable syntax CLD, i.e., that version of the Value Set Definition will not support a normative directly-computable Value Set Expansion Code Set. Of course, a different version of the Value Set Definition could specify a CLD that uses a computable syntax.

7 An HL7 Value Set Definition Expression Syntax

This section enumerates a full set of Content Expression syntax functions that can be used to define a Content Expression based on the functions used to describe the Value Set Definitions within the HL7 version 3 Model Interchange Format, Release 1[13]. This format is used for all HL7 v3 Value Set Definitions and is how these Value Set Definitions are described in the thrice-annually released MIF document. This set of Value Set Definition Content Expression functions may not be the only HL7 syntax for defining a Content Expression, but it can be considered the default syntax that may be used for Content Expressions.

---

15 http://ihtsdo.org/fileadmin/user_upload/doc/download/doc_ExpressionConstraintLanguageSpecificationAndGuide_Current-en-US_INT_20150820.pdf?ok. Accessed 2016-01-05.

16 http://www.w3.org/standards/techs/owl#w3c_all. Accessed 2016-01-05.

17 https://en.wikipedia.org/wiki/SQL. This wikipedia link serves only to clarify that use of SQL would be very specific to the implementation and while somewhat "standardized," use would not be universally interoperable.

Using this formalism, a Content Logical Definition that uses this HL7 Expression Syntax is composed of exactly one of the eight descendants listed in the Content Defining Element Types section below that specialize the abstract class *Content Defining Element* or one of its children. The HL7 CLD Expression UML model below depicts how the HL7 Expression Syntax can include ValueSetReference and CodeSystemElement that can be based on RelationshipBasedContent, CodeFilterContent that is either based on the code itself or a code directly linked to the code or a combination of these in CombinedContent.

Two of these, ValueSetReference and CombinedContent, are not expressly bound to a Code System because the internals of their definitions include such a binding. The remaining content defining classes are bound to a pair of classes that specify a Code System specification and are linked from CodeSystemElement.

A detailed set of interlinked textual examples using this HL7 Expression Syntax is provided as informative examples in the Appendix: Examples.

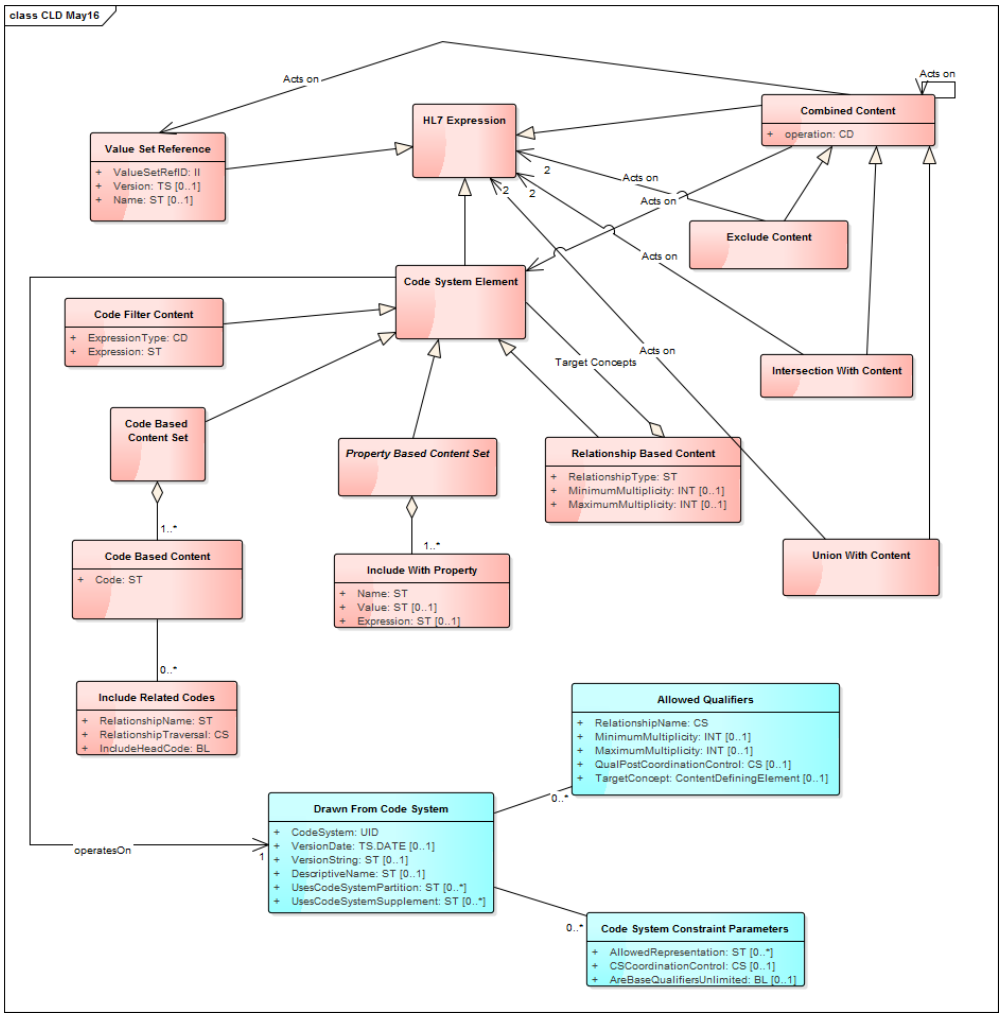

Figure 5 – UML Class Model of the HL7 Value Set Definition Expression Syntax functions described in this section that may be used in a Content Expression.

7.1.1 Content Defining Element Types

The makeup of the Content Expression may be any one of the following seven types of ***ContentDefiningElement***:

1. CodeSystemElement
    a. CodeBasedContent
    b. PropertyBasedContent
    c. RelationshipBasedContent
    d. CodeFilterContent
2. ValueSetReference
3. CombinedContent

There are four sub-types of CodeSystemElement, each of which use an aspect of the Code System noted to identify the concepts to be included in the Value Set Expansion Code Set.

The first two, CodeBasedContent and PropertyBasedContent, may describe multiple codes (for CodeBasedContent,) or multiple properties (for PropertyBasedContent) within the single CodeBasedContent or PropertyBasedContent element. These multiple items are unioned together to define the Value Set Expansion Code Set.

RelationshipBasedContent and CodeFilterContent can only include a single relationship (for RelationshipBasedContent) or code filter (for CodeFilterContent.)

Combining more than one of the sub-types of CodeSystemElement or combining a ValueSetReference with another type (including another ValueSetReference) requires the use of CombiningContent to define how the various parts are to be combined (Union, Intersection, Exclusion) as described in CombinedContent section below.

The structure and data elements of each of these types of Content Defining Element are described in detail below.

7.1.1.1 CodeSystemElement

This element (or class) is a "parent class" to five of the seven types – all except ValueSetReference and CombinedContent. Further, if this is present without any of its five subtypes (therefore only the mandatory reference to the DrawnFromCodeSystem, see below), then **the Value Set Expansion Code Set will contain all codes from the specified Code System.**

7.1.1.1.1 DrawnFromCodeSystem

Drawn from Code System is a mandatory collection of elements fully described in Source Code System Specification.

7.1.1.1.2 CodeBasedContentSet

Code Based Content Set is an aggregator that contains one or more Code Based Content elements. All Value Set Expansion Code Set members defined via individual codes or by direct subsumption are included via this element.

7.1.1.1.2.1 CodeBasedContent

CodeBasedContent designates a particular code in the Code System that anchors the content to be included in the Value Set Expansion Code Set based on codes and/or concepts related to that code (e.g., "all specializations of INT"). There may be more than one Code Based Content in a CodeBasedContentSet.

7.1.1.1.2.1.1 Code

Definition: specific Concept Representation
Description: Specifies a particular code to be used in the Value Set Expansion Code Set. It may be included on its own or it may define a header code for a set of codes based on specialization relationships with other Concept Representations in the Code System that are to be included in the Value Set Expansion Code Set. Code may be an expression using a Code System that supports the definition of concepts using expression grammar. In these situations, it should be noted that the expression must define a single concept and not be an expression that is in essence a query that can represent multiple concepts (i.e., the expression cannot include a subsumption parameter).
Usage Notes: This function requires a code anchor. Other types of relationships are handled in sections below for PropertyBasedContent and RelationshipBasedContent, so this is limited to the generalization relationship (i.e., parent/child relationships).
Data Type: ST
Cardinality: 1..1

7.1.1.1.2.1.2 IncludeRelatedCodes

The following set of elements specifies the functional behavior for the inclusion of codes in the Value Set Expansion Code Set based on relationships with the specified Code (sometimes referred to as a head code) defined in the Code System. This group repeats for each different relationship to be used.

7.1.1.1.2.1.2.1 **RelationshipName**

Definition: word or set of words to describe the relationship used for traversal to find related codes.
Description: This is the name of relationship between concepts in the Code System that can be used to identify concepts for inclusion in the Value Set Expansion Code Set. .
Usage Notes: When a Code System has a hierarchy that is not named, then RelationshipName is to be called "HIERARCHY".
Data Type: ST
Cardinality: 1..1

7.1.1.1.2.1.2.2 RelationshipTraversal

Definition: identifies which codes related by the specified relationship should be included.
Description: In a tree of codes of arbitrary depth for a particular relationship, this element identifies whether all codes walking the transitive closure of the specified relationship ("TransitiveClosure"), only those codes with a direct first-level relationship to the selected code ("DirectRelationsOnly") or only those codes walking the transitive closure of the specified relationship where the code has no outbound relationship of the specified type ("TransitiveClosureLeaves") should be included.
Usage Notes: The 3 defined values are an exhaustive list for this item.
Data Type: Constrained string (values "TransitiveClosure", "DirectRelationsOnly" or "TransitiveClosureLeaves" permitted only)
Cardinality: 1..1

7.1.1.1.2.1.2.3 IncludeHeadCode

Definition: indicates that the code is included in the content.
Description: If false, only the concepts identified via the relationship traversal based on this code are included. The CodeBasedContent.code *is not included*.
Usage Notes: The default value is TRUE. Note that if this is set to FALSE and if there are no additional codes to be included, then no codes will be included.
Data Type: BL
Cardinality: 1..1

7.1.1.1.3 PropertyBasedContentSet

This type of Content Defining Element contains elements that identify content to be included based on Code System properties held (or not held) by concepts. e.g., "all lab observation codes that are designated as 'orderable'". There is only a single repetition permitted for this Content Defining Element specification type, but it may include multiple IncludeWithProperty elements that are combined via a logical AND to generate the Concept Representations included in the Value Set Expansion Code Set.

7.1.1.1.3.1 IncludeWithProperty

Definition: indicates that codes with certain properties in the Code System are part of the Value Set Expansion Code Set.
Description: This optional repeating group of elements indicates that codes having properties in the Code System that match the specified value(s) should be considered part of the content and included in the Value Set Expansion Code Set. Multiple IncludeWithProperty instances are combined using a logical AND. This means that only codes that have properties that match **all** IncludeWithProperty instances are included in the Value Set Expansion Code Set.
Usage Notes:

Data Type: Group of elements (listed following)
Cardinality: 0..*

7.1.1.1.3.1.1 Name

Definition: word or set of words to describe the concept property.
Description: This element specifies the moniker of the named concept property in the Code System.
Usage Notes: This is the name of the property as published by the Code System publisher.
Data Type: ST
Cardinality: 1..1

7.1.1.1.3.1.2 Value

Definition: the value for which the property should be checked.
Description: The value that must be present in the name-value pair associated with the concepts to be included in the expansion code set.
Usage Notes: Either Value or Expression must be present. Note that for Code Systems that include properties without values, no value can be specified. In such a case matching on the Name alone would result in inclusion.
Data Type: ST
Cardinality: 0..1

7.1.1.1.3.1.3 Expression

Definition: a regular expression against which the property value should be evaluated .
Description: An expression that can be evaluated to determine a value that is to be found in the name-value pair associated with the concepts to be included in the expansion code set.
Usage Notes: Either Value or Expression must be present. Note that for Code Systems that include properties without values, no value can be specified. In such a case matching on the Name alone would result in inclusion.
Data Type: ST
Cardinality: 0..1

7.1.1.2 RelationshipBasedContent

This type of Content Defining Element is an element that identifies content to be included based on relationships held (or not held) by concepts, e.g., "all concepts with a 'finding site' (RelationshipType) of 'lung' (TargetConcepts) AND 'associated morphology' (RelationshipType) of 'inflammation' (TargetConcepts)".

7.1.1.2.1 RelationshipType

Definition: string uniquely identifying the relationship being used to filter the content.

Description: The string may be either a unique name or a unique identifier for the relationship type to be used.
Usage Notes: This is intended to uniquely identify the kind of relationship to be used in RelationshipBasedContent. The directionality of the relationship should also be clear, i.e., source and target.
Data Type: ST
Cardinality: 1..1

7.1.1.2.2 MinimumMultiplicity

Definition: the minimum number of the specified relationships allowed
Description: A number that indicates that concepts must have at least the specified number of relationships of the specified type meeting the constraints of the target concepts.
Usage Notes: The default value for this element is "1".
Data Type: INT
Cardinality: 0..1

7.1.1.2.3 MaximumMultiplicity

Definition: the maximum number of the specified relationships allowed
Description: A number that indicates that concepts must not have more than the specified number of relationships of the specified type meeting the constraints of the target concepts.
Usage Notes: If this is omitted, the maximum multiplicity is unbounded.
Data Type: INT
Cardinality: 0..1

7.1.1.2.4 TargetConcepts

Definition: an expression that defines the set of Concept Representations that are allowed targets of the named relationship.
Description: If present, this defines the allowed range of concepts for the named relationship.
Usage Notes: This may be specified in part as CodeBasedContent, PropertyBasedContent and/or CodeFilterContent. When more than one is used within the expression, the resulting Expansion Code Sets for each collection are unioned together.
Relationship fulfilled by: CodeSystemElement
Cardinality: 0..1

7.1.1.3 CodeFilterContent

This type of Content Defining Element is an element that identifies content to be included based on operations performed on the string value of codes, e.g., all codes starting with ‘A5’ or all codes ending with ‘B’. This content type has a single instance.

7.1.1.3.1 ExpressionType

Definition: name or description of the language in which the expression is rendered.
Description: This element contains a string that specifies the name or short description of the language used to render the expression that is used to filter the codes in CodeFilterContent.
Usage Notes: The intent of this is to eventually become a constrained list that describes the syntax used to find the needed strings using a constant name for the syntax. Given that a normative list of the string names is not known, for now an ST data type is used. A few known assumed entries are regexp[18], regexp_perl, regexp_sxp, regexp_PCRE, POSIX_BRE, POSIX_ERE and POSIX_SRE.
Data Type: ST
Cardinality: 1..1

7.1.1.3.2 Expression

Definition: the string expression against which to evaluate the code symbols.
Description: This contains the string in the syntax of the language declared in the ExpressionType (above) that is used to process the codes (strings which are Concept Representations in the code system).
Usage Notes:
Data Type: ST
Cardinality: 1..1

7.1.1.4 ValueSetReference

This type of Content Defining Element contains elements that identify content to be included that is defined by another Value Set Definition. When multiple Value Set references are needed within the HL7 Expression, each is specified within a Combined Content.

7.1.1.4.1 ValueSetRefID

Definition: the unique identifier of the referenced Value Set Definition.
Description: This is the "Value Set Identifier" of the Referenced Value Set whose Value Set Expansion Code Set is being included by reference.
Usage Notes:
Data Type: UID (OID, UUID, URI, or RUID)

---

[18] Regexp is assumed to be used as a general, non-specific syntax representation,

Cardinality: 1..1

7.1.1.4.2 Version

Definition: a string that uniquely identifies the Value Set Definition Version.
Description: This string must uniquely identify the version within the context of the Value Set Definition. It is not intended to uniquely identify the version *independent* of the Value Set Identifier. In the context of a CLD, the Version Identifier is expected to either be a string match for a specific existing Value Set Version Identifier or support a date-query based on the approach noted in Usage Notes below.
Usage Notes: While many implementers may populate a Value Set Definition Version as a date, the choice of String as a datatype supports non-date strings that some organizations may want to use to disambiguate the ordinal and sequential nature of Value Set Definition Versions without requiring a full version history. It is also recognized that many implementations will want to use a date as input to a query to retrieve the most current active Value Set Definition at that date. While a date-type version identifier simplifies the approach to meeting that requirement, matching on the version string does not guarantee identifying the correct date-based retrieval. Instead, the best approach would be to compare the requested date provided in this version string to Value Set Definition Version Activity Status Date to determine the most current active Value Set Definition Version as of the requested date.

When no Value Set Definition Version is specified, the Value Set Definition Version is inferred from the context in which the Value Set Definition is used. When the Value Set is used in an HL7 binding, then the version that is used is derived from the date/time specified in the STATIC parameter of the binding.

If the Value Set Definition is included as a Value Set Definition Reference Content Defining Element of another Value Set Definition, the version that is used is inherited from the containing Value Set Definition.

If the Value Set Definition is being expanded in the context of a dynamic binding, then the date of the evaluation of the binding (and thus the codes to be used for that conformance evaluation) is used as the Value Set Definition Version date/time.

If the Value Set Definition is not part of any binding but is being evaluated separately from its use, then the Value Set Definition Version is either the evaluation time or inherited from a containing Value Set Definition.

In essence this means that a Referenced Value Set Definition with no specified version will "inherit" the closest transitively defined version-defining date. This means that a Referenced Value Set Definition with no version ID that is referenced by two different containing Value Set Definitions, each with a different Value Set Definition Version ID, that containing a Value Set Definition Version ID will be used (as the closest transitive source of version) which can result in different versions of the same Referenced Value Set Definition being used in each of the containing value sets.

Data Type: ST
Cardinality: 0..1

7.1.1.4.3 Name

Definition: human readable string associated with the Referenced Value Set Definition.
Description: For human display only and may be any of the names that are in the Name element of the Referenced Value Set Definition.
Usage Notes:
Data Type: ST
Cardinality: 0..1

7.1.1.5 CombinedContent

This type of Content Defining Element contains elements that identify content to be included based on set operations on additional Content Expressions. The evaluation of the operations is performed in order – first unions, then intersections and then exclusions. There may be an arbitrary number of these elements of any of the three kinds in this single content specification type.

7.1.1.5.1 UnionWithContent

Definition: identifies one or more Content Defining Elements such that content from any of them is considered to be part of the Content Defining Element.
Description: This is a recursive inclusion of any sub-type of ContentDefiningElement (see Content Defining Element Types). It is intended to be additive to the set of codes that will be generated in the Value Set Expansion Code Set.
Usage Notes: The first instance of CombinedContent MUST be a UnionWithContent. The 'union' implies inclusion with any other existing parts of this Content Defining Element specification.
Data Type: ContentDefiningElement (see Content Defining Element Types)
Cardinality: 1..*

7.1.1.5.2 IntersectionWithContent

Definition: identifies Content Expressions that all concepts/codes must also match to be included.
Description: This operation permits a full Content Expression which produces a set of codes which must also be present with the codes that the other content inclusion operations produce (intersection operation).
Usage Notes: This may be employed to enable the production of the final set of codes to be populated in the Value Set Expansion Code Set rather than making use of multiple complex sets of Code Based Content inclusions.
Data Type: ContentDefiningElement (see Content Defining Element Types)
Cardinality: 0..*

7.1.1.5.3 ExcludeContent

Definition: identifies content that is explicitly excluded.
Description: Rather than specifying overly complex inclusion criteria, it is sometimes easier to implement a comprehensive inclusion Value Set Definition and then also a simply-defined set of codes that should be removed from the comprehensive Value Set Definition before being populated into the Value Set Expansion Code Set. The Content Expression provides that capability.
Usage Notes:
Data Type: ContentDefiningElement (see Content Defining Element Types)
Cardinality: 0..*

7.1.2 Source Code System Specification

The first part of the Content Specification consists of a set of parameters that lay out a series of constraints relative to Code Systems and are applied when the expressions that reference codes in Code Systems are evaluated.

Note that these parameters do not propagate across ValueSetReference instances.

7.1.2.1 DrawnFromCodeSystem

Definition: group of elements identifying the Code System that applies for this portion of the Content Logical Definition.
Description: Many of the functions in the Content Expression refer to a specific Code System. This group is not needed if the Content Expression for this portion does not refer to any Code System. Note that this is a denormalization, as this information can be ascertained by processing the Content Expression; the information is replicated here for application convenience. There may be at most one of these sections for each Content Expression section and this represents the primary Code System providing any content based upon Code Systems for this value set.
Usage Notes:
Data Type: Group of elements (listed following)
Cardinality: 0..*

7.1.2.1.1 CodeSystem

Definition: formal registered machine-processable identifier of the Code System.
Description: This is the ID of the Code System rather than its published vernacular name.
Usage Notes: This is the same identifier form that is carried in the codeSystem component or property of the HL7 v2.x and ISO 21090 coded datatypes. It should be registered in the HL7 OID Registry.
Data Type: UID (OID, UUID, URI, or RUID)
Cardinality: 1..1

7.1.2.1.2 VersionDate

Definition: a dateTime that unambiguously identifies a single published Code System version as of that dateTime.
Description: When specified, this date constrains the content to the most recent Code System version available at the specified point in time.
Usage Notes: It is expected that either the VersionDate or VersionString will be populated in DrawnFromCodeSystem, **but not both** so there is no conflict. When specified, this element is intended to allow the computable unambiguous identification of a single Code System version. The VersionDate approach **should only be used** when a VersionString cannot accurately identify the desired version. The level of precision is expected to assure identification of a single unambiguous version.
Data Type: TS.DATE
Cardinality: 0..1

7.1.2.1.3 VersionString

Definition: a string that unambiguously identifies a single Code System version at a point in time.
Description: This string is under the control of and is published by the Code System authority. When specified, it is expected to unambiguously identify the Code System at a particular published point.
Usage Notes: It is expected that either the VersionDate or VersionString will be populated in DrawnFromCodeSystem, **but not both** so that there is no conflict. When specified, this element is intended to allow the computable unambiguous identification of a single Code System version. The level of precision is expected to assure identification of a single unambiguous version.
Data Type: ST
Cardinality: 0..1

7.1.2.1.4 DescriptiveName

Definition: human readable signifier for the Code System.
Description: This is a commonly understood name for the Code System and is used for human display purposes.
Usage Notes:
Data Type: ST
Cardinality: 0..1

7.1.2.1.5 UsesCodeSystemPartition

Definition: identifies the partitions of the specified Code System that are included as part of the content.
Description: For those Code Systems that have separable partitions, this identifies a list of partitions that are included with the content.

Usage Notes: This is left unpopulated for Code Systems that do not have partitions identified by the Code System author/publisher. This should be a list of partition identifiers that must be used to correctly create a complete Value Set Expansion Code Set. If no partition(s) are listed, then the Value Set Expansion is based on all current partitions. For Value Set Definitions where conformance is a requirement, the Code System uses partitions and all current partitions may not be available to certifying agencies, the specific partitions used MUST be defined. Note that if the CLD is an Intentional definition on a Code System that has multiple partitions and this attribute is not valued, then there may be multiple valid Value Set Expansion Code Sets from the same Value Set Definition. Thus, the recommendation is to generally populate this when defining Value Sets on Code Systems that use partitions. See Code System Partitions and .
Data Type: ST
Cardinality: 0..*

7.1.2.1.6 UsesCodeSystemSupplement

Definition: identifies a collection of additional concept attributes for the Code System used as part of the Value Set Definition.
Description: Many Code Systems have supplements (see Section 7.1.2.4) published by the Code System author or by others. The functions defined in the Content Expression are permitted to refer to items that may be defined only within a supplement, as well as items defined within the base Code System, as long as the supplement(s) are listed in this element.
Usage Notes: An example of a supplement to the LOINC code system is pCLOCD, which defines additional properties for the concepts in LOINC (it extends the concept model) as well as additional concepts. See Code System Partitions and .
Data Type: ST
Cardinality: 0..*

7.1.2.2 CodeSystemConstraintParameters

Definition: contains a set of constraints on the DrawnFromCodeSystem or on the Concept Representations in that Code System.
Description: If one of the parameters is present, it affects how the Concept Representations are used in the Value Set Expansion Code Set.
Data Type: Group of elements (listed following)
Usage Notes:
Cardinality: 0..*

7.1.2.2.1 AllowedRepresentation

Definition: identifies constraints on the Concept Representations to be populated in the Value Set Expansion Code Set when the Code System has multiple Concept Representations available.

Description: If present, the constraint restricts the Concept Representations to be included in the Value Set Expansion Code Set to the specified types (e.g., 2-character vs. 3-character vs. numeric, short names vs. long names, case-sensitive vs. case-insensitive, etc.). It is unused when the code system specified has only a single Concept Representation available. If multiple Concept Representations exist and no constraints are specified, then all representations are to be included in the Value Set Expansion Code Set.
Usage Notes:
Data Type: ST
Cardinality: 0..*

7.1.2.2.2 AreBaseQualifiersUnlimited

Definition: indicates whether there are no constraints on qualifiers used in the types of Content Expressions.
Description: See Definition.
Usage Notes: If true, there are no constraints on qualifiers used in the types of Content Expressions. If false, only those qualifiers found in allowed qualifiers (7.1.2.3) with an upper cardinality greater than 0 are permitted.
Data Type: BL
Cardinality: 1..1

7.1.2.3 AllowedQualifiers

Definition: specifies, through the identified constraints, a subset of allowed concepts based on matching the specified constraints.
Description: This is a group of elements which specify various constraints to the allowed associations (or concept qualifiers) in the base concepts that are selected by the Content Logical Definition.
Data Type: Group of elements (listed following)
Usage Notes:
Cardinality: 0..*

7.1.2.3.1 RelationshipName

Definition: word or words to identify the type of relationship from the base concept to the qualifying concept.
Description: This element contains the name of types of relationships that are used for post-coordinated expressions based on relationships in the Code System.
Usage Notes:
Data Type: ST
Cardinality: 1..1

7.1.2.3.2 MinimumMultiplicity

Definition: the minimum number of the qualifiers allowed

Description: identifies the minimum number of qualifiers of this type that must be provided.
Usage Notes: Default if unspecified should be 0.
Data Type: INT (positive)
Cardinality: 0..1

7.1.2.3.3 MaximumMultiplicity

Definition: maximum number of qualifiers of this type that may be provided.
Description: Specifies whether or not there is a limit on the number of qualifiers of this specified type that are allowed.
Usage Notes: If this = "0" then the relationship is not allowed for use. Usage of this datatype should be a positive integer with the assumption that implementations need to also support a representation that can be interpreted as unlimited. If this attribute is not populated, it means it is unlimited.
Data Type: INT
Cardinality: 0..1

7.1.2.3.4 SortKey

Definition: qualifier ordinal position.
Description: Indicates where the qualifier should be included in the post-coordination syntax relative to other sibling qualifiers, if not otherwise governed by the Code System.
Usage Notes:
Data Type: INT (positive)
Cardinality: 0..1

7.1.2.3.5 TargetConcepts

Definition: identifies the set of constraints on qualifier codes.
Description: The target of a qualifying relationship which is a Content Defining Element is intended to resolve to a list of codes that act as qualifiers.
Usage Notes:
Data Type: ContentDefiningElement (recursive inclusion, see Section §6)
Cardinality: 0..1

7.1.2.4 Code System Partitions and Supplements

Code System Partitions are a unique and identifiable distinct segment of an overall Code System namespace. While most Code Systems are in essence one partition and, therefore, *do not* have identifiable partitions, some Code Systems can be made up of a collection of distinct segments. SNOMED CT is a prototypical example of such a Code System. The SNOMED CT International Edition is the base partition and can be used alone, but there are many distinct additional partitions that can be added to the international core to create what is known as "a SNOMED CT Edition". Both the international core and an edition will

have the same Code System Identifier (OID: 2.16.840.1.113883.6.96), but each unique edition can be distinguished by the partition identifiers used to create the edition. For SNOMED CT it seems likely that the "moduleID" can be used as the partition identifier. It should be noted that partitions can introduce new concepts and any other construct typical for a Code System to the resulting "complete" Code System.

Code System Supplements contain information that may (and when useful for Value Set Definitions, *do*) contain additional attributes and properties that are associated with concepts. Code System Supplements ***cannot*** add additional concepts to a code system; they may only add attributes to existing concepts within the aligned Code System. In this way a Code System Supplement is differentiated from a Code System Partition, as a partition is effectively a segment of the aligned code system namespace and *may* add additional concepts to the Code System.

Examples of a Code System Supplement include pCLOCD[19] (mentioned previously), but can also be as simple as frequency counts that describe usage statistics for concepts in a particular context. Because this supplemental information can be directly tied to individual concepts, Code System Supplements can be used as part of a Value Set Definition Content Expression to determine Value Set Expansion Code Set membership. Code System Supplements must have some sort of identifying information so that they may be uniquely identified and accessed and must identify the Code System to which they are associated.

8 Value Set Expansion File

The following material describing Value Set Expansion is provided as an INFORMATIVE supplement to the Value Set Definition.

A *Value Set Expansion Code Set* is the complete enumeration of the member list of Concept Representations as defined in a Value Set Content Logical Definition (CLD). A *Value set Expansion File* is the full set of information noted below which includes the Value Set Expansion Code Set plus the additional metadata and the additional code attributes to be included. The Value Set Expansion Code Set is the operationally useful code list artifact: it can be used to populate user interface selection tools, to validate values in a transmission and a variety of other uses. However, as noted before, the Value Set Definition which contains the CLD that determines the resulting usable code list, is different from Value Set Expansion Code Set – the set of resulting codes, which is also distinguished from the Value

[19]https://www.infoway-inforoute.ca/index.php/programs-services/standards-collaborative/pan-canadian-standards/pan-canadian-loinc-observation-code-database-pclocd-nomenclature-standard

Set Expansion File that contains not only the expansion code set but also any additional metadata and code attributes noted here. In some cases, the CLD may simply consist of an enumerated simple code list, but even in these "simple" situations the set of "usable codes" – the Value Set Expansion Code Set - may still change over time if the source system retires or deprecates concepts. And a different Expansion File can associate different code attributes to the same Expansion Code Set. Of course, this flexibility *need not be exercised!*

A Value Set Expansion Code Set will always be based on a particular (set of) Code System version(s) as indicated within the CLD or by a single *LockedDate* which will dictate the Code System version for all elements of the CLD *not* already locked in the particular CLD clause (see 5.2.3.4.1 for more details). A Value Set Expansion file will include the items noted in the sections below, but at a minimum will include the identifying Concept Representation (as described in Core Principles section 5.1.1) and the Code System Identifier for each member. It is expected that the set of concepts included will all fall within the semantic boundary of the Value Set Definition Scope textual description. Given that a Value Set Expansion Code Set is based upon a computable Value Set Definition, Value Set Expansion file content can be automatically determined at any time. In addition, a Value Set Expansion Code Set based on a specific Value Set Definition CLD may contain *different* code content then the last Value Set Expansion Code Set when the Reference Code System is updated to a new version and that update retires or adds concepts.

While the list of Concept Representations (i.e.: codes) determined by evaluation of a Value Set Definition CLD is the core of a Value Set Expansion, the Value Set Expansion file should contain more than just the Value Set Expansion Code Set to be of value. This section describes the associated metadata that clarifies the content of the Value Set Expansion file. By formally describing the Value Set Expansion file metadata, future Value Set Expansion Code Sets can be assured to have content (in addition to the required elements) that authors have determined is important for use. We use **Persistence Requirement** = Required or Optional in the following element list to indicate the elements that could be useful when stored to support consistent Value Set Expansions.

Automated generation of Value Set Expansion files will allow regimented maintenance activities such as:

a. Automatically update the Value Set Expansion file to reflect the updated content
b. Semi-automated update by having the changes be available to the maintenance author. At the maintenance author's discretion, the appropriate updates are made.
c. The update is invoked by the maintenance author, by which then the sequential step can be a. or b. above.

8.1 Metadata Needed to Describe a Value Set Expansion

A Value Set Expansion will have metadata that describes the context of creation. This specification has not fully evaluated all the potentially useful Value Set Expansion metadata. Instead the following minimal set is provided, as these are important in the description, use and maintenance of a Value Set Expansion. Future specifications will more directly address the complexities of useful Value Set Expansion content and metadata. Examples not well supported by the list provided include a method to represent Value Set member sort order, additional Code System information for display, hierarchical relationships among the expansions set concepts, etc.

8.2 Value Set Definition Identifier and Version

8.2.1 Value Set Identifier

Definition: the unique identifier of the Value Set Definition.
Description: This is the identifier for the Value Set Definition that was used to create the Value Set Expansion.
Usage Notes:
Data Type: II
Cardinality: 1..1
Persistence Requirement: Required

8.2.2 Value Set Definition Version

Definition: the specific Value Set Definition Version
Description: Since this is a specific Value Set Expansion, a reference to a specific Value Set Definition Version that was used to create the Value Set Expansion is required.
Usage Notes:
Data Type: ST
Cardinality: 1..1
Persistence Requirement: Optional

8.3 Date of Expansion

Definition: the date the Value Set Expansion was created.
Description: Date and time when the Value Set Expansion was created.
Usage Notes: This element captures only when a Value Set Expansion was created by the terminology service used to create the Value Set Expansion and, as such, **is not an identifier** for the Value Set Expansion because the content of the Value Set Expansion is dependent on the Value Set Definition Version used, the Code System Version used and may even be dependent on additional codes available for the Value Set Expansion (such as an "VSAC Expansion Profile" in which retired /inactive codes are made available for

inclusion in the Value Set Expansion Code Set.) Since expansions can be stored, the "Date of Expansion" may not be the current date. Stored Value Set Expansions are useful because many Value Set Definitions do not generate new Value Set Expansion Code Sets when Code System Versions change.

Data Type: TS.DATE
Cardinality: 1..1

8.4 Code System(s) and Version

8.4.1 CodeSystemForExpansion

Definition: group of elements identifying the Code System(s) that was used to create the Value Set Expansion Code Set.
Description: This is a repeating element that captures the enumeration of all Code Systems + versions used to create the Value Set Expansion.
Usage Notes: This is a derived set of elements based on the CLD. There will be one CodeSystemForExpansion element group for each unique combination of CodeSystem + VersionDate used to create the Value Set Expansion Code Set.
Data Type: Group of elements
Cardinality: 1..*

8.4.1.1 CodeSystem

Definition: formal registered machine-processable identifier of a Code System used to create the Value Set Expansion Code Set.
Description: This is the ID of the Code System, rather than its published vernacular name.
Usage Notes: This is the same identifier form that is carried in the codeSystem component or property of the HL7 v2.x and ISO 21090 coded datatypes. It should be registered in the HL7 OID Registry.
Data Type: UID (OID, UUID, URI, or RUID)
Cardinality: 1..1

8.4.1.2 VersionDate

Definition: a date that unambiguously identifies a single published Code System Version used to create the Value Set Expansion Code Set.
Description: The date constrains the content to the most recent Code System version available at the specified point in time. For each Code System identified, this date will be obtained from the CLD clauses that are LOCKED to indicate that a particular Code System Version was used to create the Value Set Expansion Code Set. If a CLD clause is not locked, then the VersionDate used across all non-locked clauses will be used.
Usage Notes: This element is a date and is intended to allow the computable, unambiguous identification of a single Code System Version. Given the variety and at times error-prone

approach that a Code System authority may take in crafting a version identifier for their Code System, the published Code System Identifier is not used here.
Data Type: TS.DATE
Cardinality: 1..1

8.4.1.3 VersionString

Definition: a string that identifies the Code System at a point in time.
Description: This string is under the control of and is published by the Code System authority. It is expected to unambiguously identify the Code System at a particular published point.
Usage Notes: This is included to support external references to a version of the Code System, but is not intended to be used to accurately identify a particular Code System Version. Because terminology authorities may not clearly define what string should be used to identify Code System Version, the recommendation is to use the VersionDate to determine the version expected based on the most current version as of the date specified.
Data Type: ST
Cardinality: 0..1

8.4.1.4 DescriptiveName

Definition: human readable signifier for the code system.
Description: This is a commonly understood name for the Code System and is used for human display purposes.
Usage Notes:
Data Type: ST
Cardinality: 0..1

8.4.1.5 UsesCodeSystemPartition

Definition: identifies the partitions of the specified Code System that have components that are included in the Value Set Expansion Code Set.
Description: For those Code Systems that have separable partitions, this identifies a list of partitions that are required to populate the content in the Value Set Expansion Code Set.
Usage Notes: Use of this in a Value Set Expansion is to improve clarity for implementers to understand that some of the Value Set Expansion Code Set is obtained from the partition. Inclusion of this, even if a partition is used to create the Value Set Expansion Code Set, is optional. It will also be unpopulated for Code Systems that do not have partitions identified by the Code System author/publisher. This should be a list of partition identifiers that must be used to correctly create a complete Value Set Expansion Code Set. If no partition(s) is listed, then the Value Set Expansion is based on all current partitions. For Value Set Definitions where conformance is a requirement the Code System uses partitions and all current partitions may not be available to certifying agencies, the specific partitions used MUST be defined.

Data Type: ST
Cardinality: 0..*

8.4.1.6 UsesCodeSystemSupplement

Definition: identifies the Code System Supplements of the specified Code System that have components that are included in the Value Set Expansion Code Set.
Description: Many Code Systems have supplements published by the Code System author or by others. The functions defined in the Content Logical Definition are permitted to refer to items that are defined within a supplement, as well as within the base Code System as long as the supplement(s) are listed in this element. As in the CLD, this element in a Value Set Expansion clarifies that supplement content is used in the Value Set Expansion Code Set. Inclusion of this, even if a supplement is used for the expansion, is optional.
Usage Notes: An example of a supplement to the LOINC code system is pCLOCD, which defines additional properties for the concepts in LOINC (it extends the concept model).
Data Type: ST
Cardinality: 0..*
Persistence Requirement: Optional (This is included because a Code System Supplement may be used to add additional concept-level information (information that is *not* available from the Code System) to the Value Set Expansion file, but that supplement may *not* have been used in the CLD therefore will not be identified by reviewing the CLD requirements).

8.5 Expansion Steward

Definition: the entity that is responsible for creating the Value Set Expansion.
Description: This is a textual description of the organizational entity responsible for the Value Set Expansion and is responsible for the maintenance and changes. The information included should include contact information.
Usage Notes:
Data Type: ST
Cardinality: 0..1

8.6 Expansion Concept Attributes

Definition: textual description of the concept-based attributes included in the Value Set Expansion.
Description: This is a textual description of the concept attributes included in the Value Set Expansion. In essence this is the list of the headers in a columnar display of the Value Set Expansion Code Set. Each concept attribute included in the Value Set Expansion Code Set should be described. Concept attributes are expected to be derived from the Code System or from a concept-level Code System Supplement.

Usage Notes: The goal is to have self-describing data; therefore, this metadata element should be a formalized list of the Code System or Code System Supplement attributes included in the Value Set Expansion file to clarify what is included in each entry of the Value Set Expansion Code Set. It is expected that this list must include the minimum set of Code System attributes necessary to describe the Concept Representations included in the Value Set Expansion Code Set, including but not limited to Concept Identifier, Code System Identifier and Code System Version. In addition the formal Code System attribute names (and identifiers, if available) for all Code System attributes should be included, as well as specific characterizations of any Code System Supplement information included. This is critical so that users of the Value Set Expansion can understand the information in the Value Set Expansion. This should also explain the format of the file. Where available, the text should use formal Code System attribute name or identifiers for the attributes used.
Data Type: ST
Cardinality: 0..1
Persistence Requirement: Optional

8.6.1 SortKey

Sorting the display order of the included concepts in a Value Set Expansion Code Set is commonly useful. SortKey is used to capture this information. It is currently used in the MIF structures for HL7 Value Sets. Before Value Set Expansion Code Sets were clearly separated from a Value Set Definition, the SortKey attribute was included in the Value Set elements. Further evaluation of this clarified that SortKey is a Value Set Expansion attribute.

Definition: indicates the relative order in which the listed code should be presented.
Description: This controls the ordering of the codes in the Value Set Expansion Code Set. Note that it is only relevant if Code is specified and if all other Content Defining Element Types are CodeBasedContent.
Usage Notes:
Data Type: INT
Cardinality: 0..1

8.7 Expansion Identifier

Definition: a string that can be used to identify the Value Set Expansion Code Set.
Description: A string that when combined with the Value Set Definition Identifier and Version uniquely identifies the Value Set Expansion Code Set.
Usage Notes: Because this identifier may only be unique within the context of a single terminology service, it is recommended that a URI be used so that the identifier is globally unique.
Data Type: ST
Cardinality: 0..1

9 Implementation Considerations

Value Sets have been a part of model constructs from the very beginning, but in general have always been implemented in a way to support easy use and review of the model content. In most cases a Value Set has either been a list of ideas rendered within the structure of the model or as an attached supplement. The format of these artifacts is usually a table, because at a minimum the members of the Value Set are linked tuples with some sort of code/mnemonic linked to a more wordy description with the expectation that the code/mnemonic was used in model instances with the description displayed for human use. The table would have some sort of identifier (often derived by the structure of the publishing document). It is very likely that there are more Value Sets developed, stored and maintained in spreadsheet format than in any other technology. As a quick and efficient way of getting the material in a presentable form, this has worked for years. In fact, in a very real way the HL7 V2.x world has continued to struggle to move beyond this approach.

We now understand that such embedded "Value Set" artifacts are problematic because they live in isolation with little overarching governance. At a minimum there is value in separating the Value Set content from the artifacts using that content to encourage re-use in other appropriate models. This specification describes the metadata that should be used to describe a Value Set that stands alone and as such can be maintained, versioned, vetted and reused. As such, that definition can then be applied to a Code System to produce an instance Value Set Expansion file that includes the Value Set Expansion Code Set of member concepts. The now independent Value Set Definition can be repeatedly applied to every new version of the Code System, producing the latest up-to-date Value Set Expansion Code Set for each new Code System version.

9.1 Implementation Technologies

Traditionally Value Sets have been managed in files, usually spreadsheets. This technology is still useful and can contain all the information noted in the Value Set Definition specification. Obviously it can also be used for a Value Set Expansion (as this technology has essentially been the method of choice in most systems), but new technologies that provide database support to the Value Set metadata are extremely useful for repositories, particularly those with authoring and content delivery functionality. This specification is agnostic to the technology choices that may be used to implement the requirements.

9.2 Authoring

The importance of the provenance in support of the creation and maintenance of Value Sets is of utmost importance to vetting and open reuse of Value Set Definitions. Therefore, a

number of workflow-related elements are included in this specification. For some, authoring is a complex, multi-step process that must support internal (and potentially external) review, approvals, versions and publication. Only through this kind of rigorous process will the use of understandable and consistent Value Sets be possible, a key milestone in true interoperability. As noted in the specification, Value Sets can change stewardship because many useful Value Sets are developed under a contractual relationship that can change over time.

9.3 Reuse of Value Sets

Consistency in meaning across implementations is the cornerstone of sematic interoperability. Value Set reuse is critical to this process and through reuse and re-vetting, improvements are inevitable. Implementers do not want multiple Value Sets with small variations (or worse, no variation) – reuse means a common approach to a single issue and provides for meaningful variation where required.

9.4 Distribution

The Value Set Definition specification has as a primary goal support for alignment of the critical elements needed to describe a Value Set, a defined approach to versioning and a (hopefully) complete set of functions needed to describe a simple or complex query into a Code System to retrieve the Value Set Expansion that provides the actual Value Set members. While *this* version of the specification does not describe a syntax for exchange and distribution of either the Value Set Definition or the Value Set Expansion, it does provide a platform to determine this in the future.

9.5 Impact of Code System Evolution

A central tenet to the specification is the fact that Value Set Definitions are often created with the expectation that the information is robust and will stand "the test of time" – most importantly, it will continue to generate appropriate Value Set Expansions as the Code System matures. This means that a single Value Set Definition version that is not tied to a particular Code System Version is expected to continue to produce appropriate and usable Value Set Expansion Code Sets for every new Code System version. Authors/stewards may review these new Value Set Expansions to be assured that assumption remains correct. If they find problematic changes in the Value Set Expansion Code Set based on a new Code System version, they can make appropriate changes to the CLD and publish this (as required) as a new Value Set Definition Version. They may even restrict the use of the prior Value Set Definition Version such that new content based on that outdated Value Set Definition is not allowed, forcing migration to the newer version (when using the appropriate Code System version).

10 Relationships to Other HL7 Standards

The following material describing the relationship of the Value Set Definition standard to other HL7 standards is provided as an INFORMATIVE supplement.

10.1 Version 3

The HL7-defined Value Sets are specified in the HL7 Vocabulary section of the HL7 V3 Normative Edition (Foundation Chapter). A detailed description of Value Sets (HL7 and other) and their relationships to the other aspects of the V3 standard is provided in *Core Principles and Properties of HL7 Version 3 Models.*

In HL7 V3, a Value Set (a collection of concepts drawn from one or more Code Systems grouped together for a specific purpose) is one of the key defined vocabulary structures, typically associated with a coded Model Element (in a message or document) through a vocabulary binding from its associated Concept Domain (for Context binding) or directly from the Model Element (for Model binding). In addition to the descriptions in Chapter 5 of this document, the V3 vocabulary structures and their usage is described in further detail in *Chapter 5 - Coded Model Elements and their Vocabularies* of *Core Principles and Properties of HL7 Version 3 Models*. Section 5.1.3 of *Core Principles…* describes Value Sets and their attributes and associated rules (the descriptions in *Core Principles…* should be in alignment with the specifications in this document – and, where not, this is expected to be addressed in an upcoming *Core Principles…* release). Section 5.2 discusses Vocabulary Conformance. Section 5.3 provides a detailed discussion of Vocabulary Binding, including the definitions, specifications and rules for Model and Context (i.e. realm-specific) Binding.

The HL7 Vocabulary includes a large number of (> 1800) HL7-defined Value Sets – the detailed contents and descriptions are found in the HL7 V3 Normative Edition. The "source of truth" for the HL7 Vocabulary (including the value sets) is the MIF (viewable using the RoseTree tool). In addition to the HL7 Value Sets, there are many other Value Sets defined by organizations outside of HL7 that are not represented or referenced directly in the MIF or Normative Edition, but still may be used in HL7 messages and documents. The same HL7 vocabulary binding machinery is also used for specifying these "non-HL7" Value Sets in message and document instances.

The Data Types: Abstract R2 specification (used in current versions of the HL7 V3 RIM) contains references to Value Sets. The CD (Concept Descriptor) data type (and its CV flavor) contains properties for valueSet and valueSetVersion, which are used to specify the Value Set and Version that applied when the specific *CD* instance was created. It should be noted that the earlier Data Types: Abstract (R1) specification **does not** explicitly reference Value Sets in the CD data type (or elsewhere). This has potentially important implications,

because the earlier R1 data types are still being widely used in the CDA R2 standard (see the CDA discussion in 11.4).

The R2 CD valueSet property is a UID (UniqueIdentifierString), which is intended to identify an object "in a globally unique and timeless manner", and may be populated with either an OID or UUID. This should always be populated with the Value Set Identifier.

The R2 Data Types specification also requires that the version of the Value Set SHALL also be provided. The valueSetVersion property is a string (ST.SIMPLE), and its value "must properly identify a particular version of the Value Set following the rules defined by the Value Set or its publisher." For HL7 defined Value Sets, the version SHALL be the date/time that the Value Set Definition was published in a ballot. This date will match the expectation that the valueSetVersion property be populated with the Value Set Expansion: Date of from the Value Set Expansion Code Set used to populate the code component of the CD.

10.2 Version 2

HL7 Version 2 mentioned Value Sets in various descriptions for many years, but did not formally define them or their explicit reference. In V2.5.1, the CWE datatype states:

> The CWE data type should be used for coded fields that are optional or where it is permissible to send text for items that are not yet a part of the approved value set. In the normal situation, the identifier is valued with the code from the value set. If the value of the field is known, but is not part of the value set, then the value is sent as text, and the identifier has no value.

In addition, Value Sets are mentioned in the second component of the SPS datatype, where the values are drawn from HL7 table 0371:

> The table's values are taken from NCCLS AUTO4. The value set can be extended with user specific values.

In the discussions on V2 Conformance (section 2.12 in V2.5.1), Value Sets are mentioned as being part of the Static Definition of a Conformance Profile. In version 2.6, additional text was added in the definition of the CWE datatype to further describe how Value Sets are used in conjunction with the V2 coded datatypes.

The explicit reference to Value Sets and their identifiers was introduced in version 2.7, when three pairs of components were added to the CWE and CNE datatypes. Each pair of components holds the Value Set OID (carried as a string) and the Value Set Version ID, which is a Date/Time. Note that the Value Set Version ID components are defined as **"the date is the date/time that the value set being used was published."** This should be populated with the Value Set Expansion: Date of Expansion for the Value Set Expansion file from which the code populated in the code component was drawn.

10.3 Model Interchange Format

HL7's Model Interchange Format (MIF) originally defined most (though not all) of the content formal Value Set structure described in this document, including the formal definition, support for partitions, Code System Supplements, control over post-coordination, etc. As such, MIF is a conformant implementation of this specification.

MIF has been exercised by HL7 as the primary publication mechanism (and tooling support mechanism) for HL7 V3 vocabulary maintained by HL7 International for the past 10 years. It is the format consumed by tooling such as the V3 Generator and RoseTree. It has also been used by at least some HL7 affiliates, meaning that much of the content documented in this specification has been subject to implementation experience and stems from real-world use cases.

Because MIF is used specifically for HL7 use-cases, there are some constraints on what it can do. Examples include:

- No support for version identifiers other than dates or date-time values
- No support for multiple Value Set Expansions, maintaining Value Set Expansions or the level of detail supported in this specification for Value Set Expansions.

A couple of minor changes to align with this specification will be performed once this specification passes Normative ballot.

10.4 CDA

Clinical Document Architecture Release 2 (CDA R2) is one of the HL7 V3 family standards; however, CDA R2 uses Release 1 of the Data Types –Abstract specification, which **does not** include explicit references to Value Sets and versions in the CD (Concept Descriptor) or other coded data types. In particular implementation guides where references to Value Sets are required, such as QRDA (HL7 Implementation Guide for CDA® Release 2: Quality Reporting Document Architecture – Category I), it may be necessary to use a CDA extension in order to represent this data – for QRDA, this extension is the sdtc:valueSet attribute (note that there is no extension currently being used in QRDA to represent the value set **version**).

The CDA R2 base standard includes the specification of a large number of Value Sets that are defined in tables contained within the CDA R2 standard document itself (HL7 Clinical Document Architecture, Release 2.0). The majority of these Value Sets are enumerations (extensional definitions), but some are defined by rather simple intentional rules – an example is "Table 31: Value set for RelatedEntity.classCode (CNE)", which is defined as "all subtype of RoleClassMutualRelationship".

10.5 Fast Health Interoperable Resources

Fast Health Interoperable Resources (FHIR) is the newest HL7 specification for data exchange. It includes a specific ValueSet resource that corresponds to much of the content found here. The ValueSet resource is a first-order structure like any other resource and can be conveyed in instances along with clinical data and can be queried and maintained using FHIR RESTful services.

The design of the core ValueSet resource is limited to those features and capabilities expected to be used by most systems, so a considerable amount of the capability found in this specification is not supported directly by the resource. However, FHIR makes use of a profiling structure in which extensions can add capabilities (and constraints) not found in the base resource. A Profile that strictly aligns with the capabilities defined in this document will be developed during the STU phase of this specification to provide guidance to FHIR implementers who wish to create more sophisticated value set definitions.

FHIR's ValueSet goes beyond the capabilities documented here, in that ValueSet can also be used define code systems "on the fly". This is done as an implementer convenience as frequently Code Systems and Value Sets are created on a 1..1 basis for things like answers to questions in questionnaires, for structural codes and other purposes. This behavior falls outside of the scope of this document as formally, Value Set Definitions do not include the definition of Code Systems, Code System Supplements, etc.

10.6 Common Terminology Services 2

The Common Terminology Services – Release 2 (CTS 2) specification[20] is intended to mediate among disparate terminology sources by defining a common information model and computational model. The information model specified in the CTS 2 Platform Independent Model (PIM) outlines the structural definition, attributes and associations of the elements common across structured terminologies and terminology elements such as Value Sets. CTS 2 does not mandate specific terminology components, but provides the infrastructure to support a diverse array of structured terminology representations.

In the future, when an individual with sufficient knowledge of CTS 2 can provide the necessary support, the metadata elements specified in this document will be mapped to the CTS 2 PIM. There is no evidence that there will be significant difficulty in defining how HL7

---

[20] OMG CTS2 CTS2 Value Set Services 1.1 Section 4.3 appears to align with the CLD. See http://www.omg.org/cgi-bin/doc?formal/13-12-08.pdf

Value Sets would be represented in CTS 2 in accordance with the Value Set Definition Project.

11 Appendix

11.1 Examples

Example value set content based on the HL7 Expression syntax, rendered within Microsoft Excel workbooks, are available for download at this url:

http://wiki.hl7.org/index.php?title=HL7_VSD_Expression_Syntax

Additional examples will be provided at this location over time.

Currently the following two primary example groups are provided:

11.2 Chlamydia Value Sets

The Chlamydia value sets described in these examples are actual value sets used in the US electronic quality measure program. The "Grouping" example and the three grouped value sets that make up the grouping: Chlamydia ICD9, Chlamydia ICD10, and Chlamydia SCT, all exist in the US NLM Value Set Authority Center (VSAC). The Chlamydia Union example demonstrates how a single value set using the HL7 Expression could be constructed using enumerated codes from multiple code systems, with each CodeBasedContent set unioned together. Both the Chlamydia Union definition and the Chlamydia Grouping definition generate *the same Expansion Code Set.*

11.3 RoleClass-based Value Sets

The RoleClassAssignedEntity Value Set uses the RoleClass code system. This example demonstrates two things:

1. Both RoleClassAssignedEntity and RoleClassContact use an "intentional" definition style to identifying concepts to be included in the Expansion Code Set wherein a single code is provided along with a relationship traversal that clarifies that a transitive closure traversal is to be followed to find all codes related to the code provided. This is how "and all descendants" is communicated.
2. RoleClassAssignedEntity unions together the "code and all descendants" noted above plus a reference to another value set - RoleClassContact.

