LexEVS 6.0 Design Document - Detailed Design - Value Set Definition

Contents of this Page

Document Information

Author: Craig Stancl, Sridhar Dwarkanath
Email: stancl.craig@mayo.edu, dwarkanath.sridhar@mayo.edu
Team: LexEVS

Contract: CBITT BOA Subcontract# 29XS223
Client: NCI CBIIT
National Institutes of Heath
US Department of Health and Human Services

Revision History

Version	Date	Description of Changes	Author
1.0	5/14/10	Initial Version Approved via Design Review	Team

Value Set Definition

Value Set Definition with in the LexGrid logical model defines the contents of Value Set. The contents are concept codes defined in referencing Code System. Value Set can contain concept codes from one or more Code Systems.

Compared to ISO 11179 model

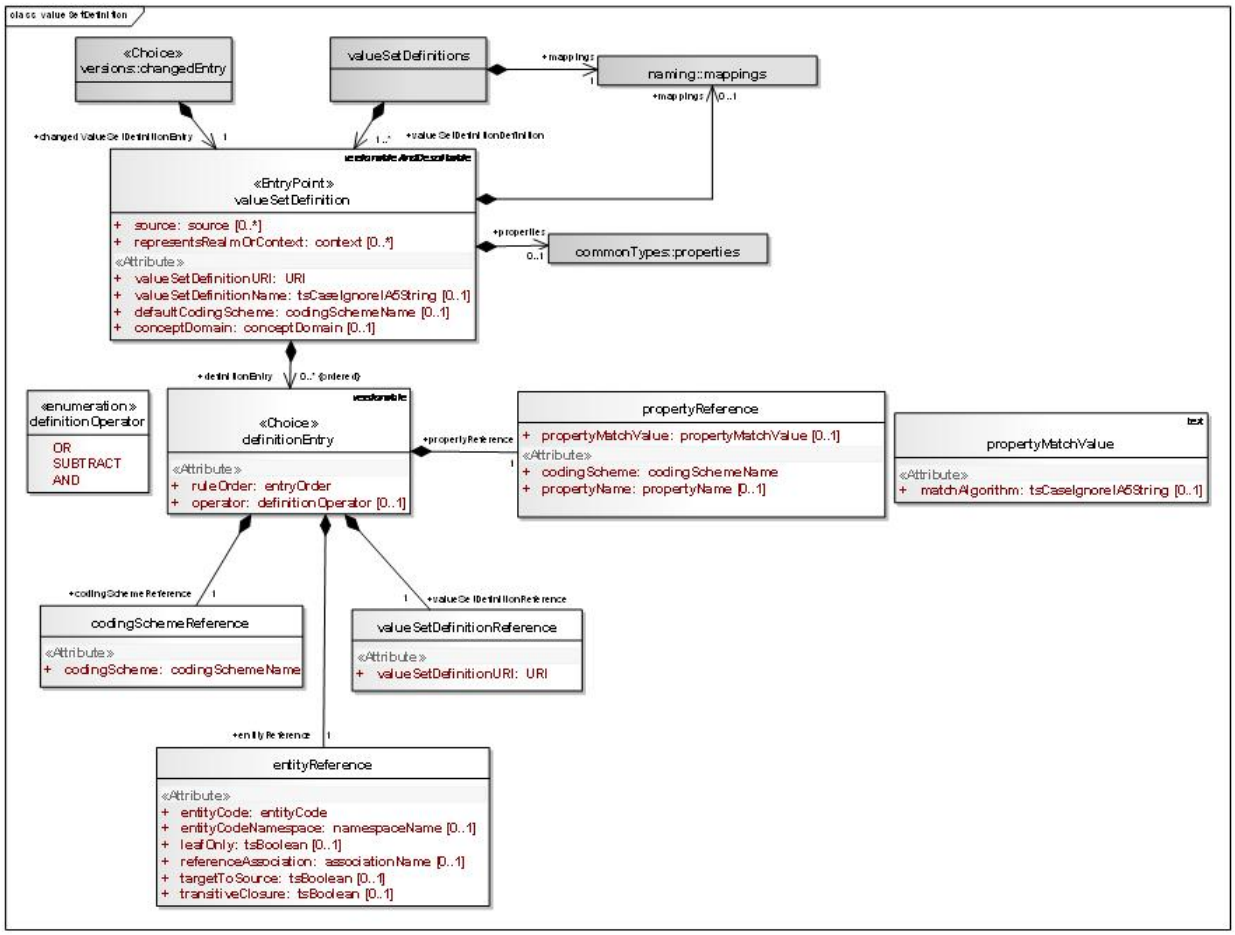
The LexGrid "value set definition" is analogous to the 11179 enumerated conceptual domain. We support the notion of an intrinsically defined enumerated domain, so the 11179 enumerated conceptual domain is really the output of the resolve value set definition function.

Value meanings are identified by entity codes within a given code system. The mapping between permissible values and value meanings is currently accomplished via a mapping association, where we treat the set of permissible values as a "mini code system". An example of how this might work is that HL7 has a set of permissible values of "M", "F", "U", which are in the administrative gender "code system". In the pure HL7 context, these might well map to the equivalent value meanings identified by "M", "F" and "U". If, however, we were using, say, the UMLS as a source of meaning, the same codes would map to the corresponding UMLS CUIs.

Compared to HL7 Value Set

The term "value set", when discussed in the context of HL7 can be somewhat ambiguous. There are at least three related artifacts that are sometimes called "value sets", including 1) a definition or algorithm that, when interpreted, produces a set of concept codes, 2) the actual set of concept codes that result from the execution of a the definition or algorithm, and 3) a subset of this set of concept codes coupled with appropriate designations and identifying information. For this reason, the LexEVS model uses the names "value set definition", "value set resolution" and "pick list definition", "pick list resolution" respectively to represent the three different senses of "value set" described above.

Value Set Definition logical model



Value Set Definition logical model description

Value Set Definition

A definition of a given value set. A value set definition can be a simple description with no associated value set entries, or it can consist of one or more definitionEntries that resolve to an enumerated list of entityCodes when applied to one or more codingScheme versions.

Attributes of Value Set Definition:
Source: The local identifiers of the source(s) of this property. Must match a local id of a supportedSource in the corresponding mappings section.
representsRealmOrContext: The local identifiers of the context(s) in which this value set applies. Must match a local id of a supportedContext in the corresponding mappings section.
valueSetDefinitionURI: The URI of this value set definition.
valueSetDefinitionName: The name of this value set definition, if any.
defaultCodingScheme: Local name of the primary coding scheme from which the set is drawn. defaultCodingScheme must match a local id of a supportedCodingScheme in the mappings section.
conceptDomain: Local name of the concept domain. When present, the contents of value set are considered to be bound to this specific concept domain. conceptDomain must match a local id of a supportedConceptDomain in the mappings section.

Value Set Definition Entry

A reference to an entry code, a coding scheme, entity code property name or value, or another value set definition along with the instructions about how the reference is applied. Definition entries are applied in entryOrder, with each successive entry either adding to or subtracting from the final set of entity codes.

Attributes of Value Set Definition Entry :
ruleOrder: The unique identifier of the definition entry within the definition as well as the relative order in which this entry should be applied
operator: How this entry is to be applied to the value set

Coding Scheme Reference

A reference to all of the entity codes in a given coding scheme.

Attributes of Coding Scheme Reference :
codingScheme: The local identifier of the coding scheme that the entity codes are drawn from . codingSchemeName must match a local id of a supportedCodingScheme in the mappings section.

Value Set References

A reference to the set of codes defined in another value set definition.

Attributes of Value Set Reference :
valueSetDefinitionURI: The URI of the value set definition to apply the operator to. This value set may be contained within the local service or may need to be resolved externally.

Entity Reference

A reference to an entityCode and/or one or more entityCodes that have a relationship to the specified entity code.

Attributes of Entity Reference :
entityCode: The entity code being referenced.
entityCodeNamespace: Local identifier of the namespace of the entityCode. entityCodeNamespace must match a local id of a supportedNamespace in the corresponding mappings section. If omitted, the URI of the defaultCodingScheme will be used as the URI of the entity code.
leafOnly: If true and referenceAssociation is supplied and referenceAssociation is defined as transitive, include all entity codes that are "leaves" in transitive closure of referenceAssociation as applied to entity code. Default: false
referenceAssociation: The local identifier of an association that appears in the native relations collection in the default coding scheme. This association is used to describe a set of entity codes. If absent, only the entityCode itself is included in this definition.
targetToSource: If true and referenceAssociation is supplied, navigate from entityCode as the association target to the corresponding sources. If transitiveClosure is true and the referenceAssociation is transitive, include all the ancestors in the list rather than just the direct "parents" (sources).
transitiveClosure: If true and referenceAssociation is supplied and referenceAssociation is defined as transitive, include all entity codes that belong to transitive closure of referenceAssociation as applied to entity code. Default: false

Property Reference

A reference to a propertyName or propertyValue and matchAlgoritm to use.

Attributes of Property Reference :
codingScheme: The local identifier of the codingScheme that this propertyReference will be resolved against. codingScheme must match a local id of a supportedCodingscheme in the corresponding mappings section.
propertyName: The local identifier to be used to restrict the entities to have property with this name. Must match a local id of a supportedProperty in the corresponding mappings section.
propertyMatchValue: Value to be used to restrict entity property. matchAlgorithm can be used in conjunction to get matching entity properties.

Property Match Value

Property match value to be used to restrict entity property. matchAlgorithm can be used in conjunction to get matching entity properties.

Attributes of Property Match Value :
matchAlgorithm: Algorithm to be used in conjunction with propertyValue.

Definition Operator

The description of how a given definition entry is applied.

Attributes of Definition Operator :
OR: Add the set of entityCodes described by the currentEntity to the value set. (logical OR)
SUBTRACT: Subtract (remove) the set of entityCodes described by the currentEntity to the value set. (logical NAND)
AND: Only include the entity codes that are both in the value set and the definition entry. (logical AND)

Possible forms of Value Set Definitions

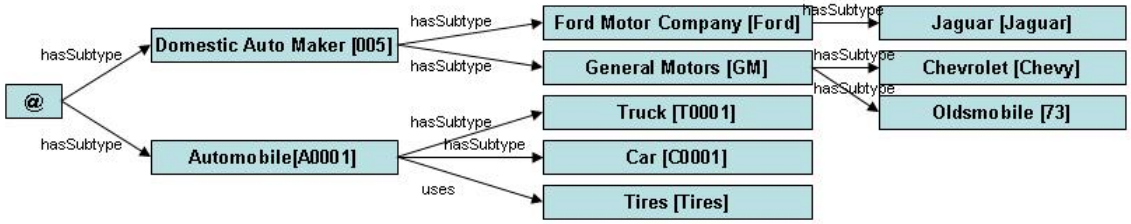
- Definition containing just the reference to Code System - includes all the concept codes in the referencing code system.
- Definition containing just the reference to other Value Set Definition -includes all the concept codes defined in the referencing Value Set Definition.
- Definition containing reference to Code System plus concept codes - includes individual concept codes defined in the definition from the referencing code system.
- Definition containing reference to Code System plus concept codes plus relationship plus additional rules (leaf only, immediate children, matching property name/value etc) - includes concept codes from the referencing code system that satisfies the rule set defined in the definition.
- Combination of any of the above with OR/AND/SUBTRACT operations.

Definition of the Value Set could be as simple as specifying just individual concept codes or specifying to include all the children of concept 'Body Structure' from Code System 'SNOMED CT' to complex definition containing multiple rule sets.

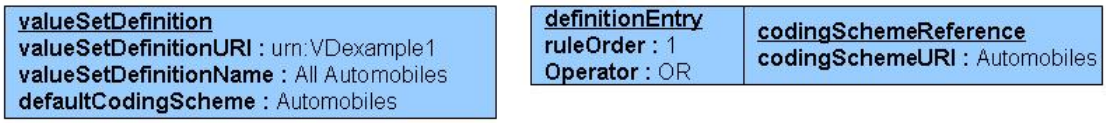
Fictitious Examples of Value Set Definitions

Example 1

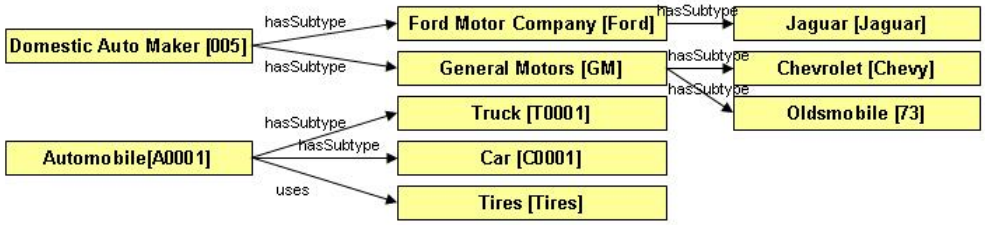
Say we have a Code System 'Automobiles' that contains following concept codes:



So to define Value Set Definition to include all the concept codes in Code System 'Automobiles'. We could define Value Set Definition as simple as:



And when this definition is resolved, we get back following concept codes:



Example 2

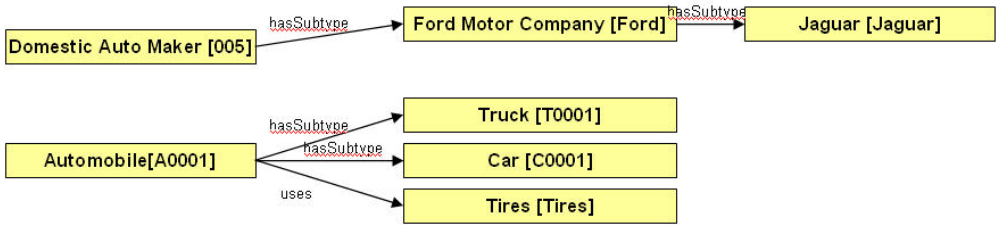
Using the same Code System 'Automobiles', if we want to define Value Set Definition to include all concept codes except 'General Motors' and its children, we could define it some thing like the following using two definition entries:

valueSetDefinition
valueSetDefinitionURI : urn:VDexample2
valueSetDefinitionName : All Automobiles BUT GM
defaultCodingScheme : Automobiles

definitionEntry	codingSchemeReference
ruleOrder : 1	**codingSchemeURI :** Automobiles
Operator : OR	

definitionEntry	entityReference
ruleOrder : 2	**entityCode :** GM
Operator : SUBTRACT	**referenceAssociation :** hasSybtype
	transitiveClosure : true

And when this definition is resolved, we get back following concepts:



Value Set Definition Resolution

- A value set definition has to be made against a specific version of a code system.
- But it doesn't have to be resolved against the same version.
- Even a simple list (a, b, c, d) needs to be resolved as, at some future date, "c" might be retired.
- Resolution does not create static artifact.

Value Set Definition Versions

Value Set Definitions are versioned. The version of Value Set Definition changes when ever the definition is changed, it could be adding or removing Code System reference or changes in the rule set.

Value Set Services

LexEVS Value Set Services is an integral part of LexEVS Core API. It provides:

- Ability to load Value Set Definitions into LexGrid repository
- Ability to apply user restrictions and dynamically resolve the definitions during run time
- Ability to author Value Set Definition

Major functions provided by Value Set Definition Services

- **Administration functions:**
  - Ability to load Value Set Definitions into LexGrid repository.
  - Ability to remove Value Set Definitions from LexGrid repository.
  - **Here are some of the admin methods available:**
    - **loadValueSetDefinition(ValueSetDefinition vddef, String systemReleaseURI)** - Loads supplied valueSetDefinition object
    - **loadValueSetDefinition(InputStream inputStream, boolean failOnAllErrors)** - Loads valueSetDefinitions found in inputStream
    - **loadValueSetDefinition(String xmlFileLocation, boolean failOnAllErrors)** - Loads valueSetDefinitions found in input xml file
    - **removeValueSetDefinition(URI valueSetDefinitionURI)** - Removes supplied value set definition from the system
    - **removeAllValueSetDefinitions()** - Removes all value set definitions from the system
- **Query functions:**
  - Ability to list all the Value Set Definitions loaded in the system.
  - Ability to dynamically resolve Value Set Definition with/without user supplied restrictions.
  - Ability to check if a concept code is part of given Value Set.
  - Ability to check if one Value Set is sub set of other Value Set.
  - **Here are some of the query methods available:**

- **isConceptInSet(String entityCode, URI valueSetDefinitionURI)** - Determine if the supplied entity code is a valid result for the supplied value set and, if it is, return the particular codingSchemeVersion that was used
- **isConceptInSet(String entityCode, URI entityCodeNamespace, CodingSchemeVersionOrTag csvt, URI valueSetDefinitionURI)** - Similar to previous method, this method determine if the supplied entity code and entity namespace is a valid result for the supplied value set when resolved against supplied Coding Scheme Version or Tag
- **resolveValueSetDefinition(URI valueSetDefinitionURI, AbsoluteCodingSchemeVersionReferenceList csVersionList)** - Resolve a value set using the supplied set of coding scheme versions
    - **ResolvedValueSetDefinition :** A resolved Value Set Definition containing the Coding Scheme Version reference list that was used to resolve the Value Set and an iterator for resolved concepts
- **getValueSetEntitiesForTerm(String term, URI valueSetDefinitionURI, String matchAlgorithm)** - Resolves the value set supplied and restricts to the term and matchAlgorithm supplied
    - **ResolvedValueSetCodedNodeSet :** Contains the codingScheme URI and Version that was used to resolve and the CodedNodeSet. Note : the CodedNodeSet is unresolved
- **isSubSet(URI childValueSetDefinitionURI, URI parentValueSetDefinitionURI)** - Check whether childValueSetDefinitionURI is a child of parentValueSetDefinitionURI
- **getValueSetDefinition(URI valueSetDefinitionURI)** - Returns value set definition for supplied value set definition URI
- **listValueSetDefinitions(String valueSetDefinitionName)** - Return the URI's for the value set definition(s) for the supplied name
- **getAllValueSetDefinitionsWithNoNames()** - Return the URI's of all unnamed value set definition(s)
- **getCodingSchemesInValueSetDefinition(URI valueSetDefinitionURI)** - Returns list of coding scheme summary that is referenced by the supplied value set definition URI
- **isValueSet(String entityCode, String codingSchemeName, CodingSchemeVersionOrTag csvt)** - Determine if the supplied entity code is of type valueSet in supplied coding scheme and, if it is, return the true, otherwise return false

- **Authoring functions:**
    - **valueSetDefinition** and **definitionEntry** classes are versionable entities in LexGrid model. Following authoring functions can be performed on them:
        - **NEW :** Create new Value Set Definition or add new defintionEntry to existing Value Set Definition.
        - **MODIFY :** Modify existing attributes of Value Set Definition or definitionEntry.
        - **REMOVE :** Ability to completely remove Value Set Definition or definitionEntry. This is not the same as deprecated, as the entity ceases to exist in future versions.

Refer to Revision section of this document for information on how revisions can be applied and how LexEVS handles them.

CTS 2 specific Value Set Services

CTS 2 specification uses term "Value Set" which is basically LexEVS implementation of "Value Set Definition". Though, CTS 2 implementation functions uses term "Value Set", internally, the LexEVS Value Set Definition Services are used. CTS 2 specification contains several Value Set specific profiles as described in the LexEVS 6.0 Design Document - Solution Architecture section. All of these functions can be called using CTS 2 interface as described in the LexEVS 6.0 Design Document - Solution Architecture section.

Here are CTS 2 profiles specific to Value Sets:

- **Administration profile:**

Function	Definition
Import Value Set Version	Ability to import values sets

- **Query profile:**

Function	Definition
List Value Sets	The ability to determine what value sets are available to a Terminology Service. This includes seeing a listing of the available value sets that match some search criteria, as well as the details pertaining to each value set available to the terminology service.
Return Value Set Details	The ability to retrieve a specific value set, with associated attributes and other metadata.
List Value Set Contents	The ability to see a listing of specific concepts, as well as the details pertaining to each concept in any of the given value sets available to a terminology service.
Check Concept Value Set Membership	The ability to validate that a given concept exists in a given value set.
Check Value Set Subsumption	Determine whether one of the two supplied value sets subsumes the other

- **Authoring profile:**

Function	Definition
Create Value Set	The ability to create a dynamic value set that is defined by a computable expression that can be resolved to an exact list of coded concepts at any given point in time.
Maintain Value Set	Update properties or expression of a value set definition (extensional and intensional value sets).

Update Value Set Status	The ability to modify the status of a value set.

Load Scripts

Scripts to load Value Set Definitions into LexEVS system will be located under 'Admin' folder of LexEVS install directory. This loader scripts will only load data in XML file that is in LexGrid format.

Value Set Definition Loader

LoadValueSetDefinition.bat for Windows environment and **LoadValueSetDefinition.sh** for Unix environment.
Both these scripts takes in following parameters :

-in	Input <uri> URI or path specifying location of the source file.
-v	Validate <int> Perform validation of the candidate resource without loading data. Supported levels of validation includes : 0 = Verify document is well-formed 1 = Verify document is valid

Example:

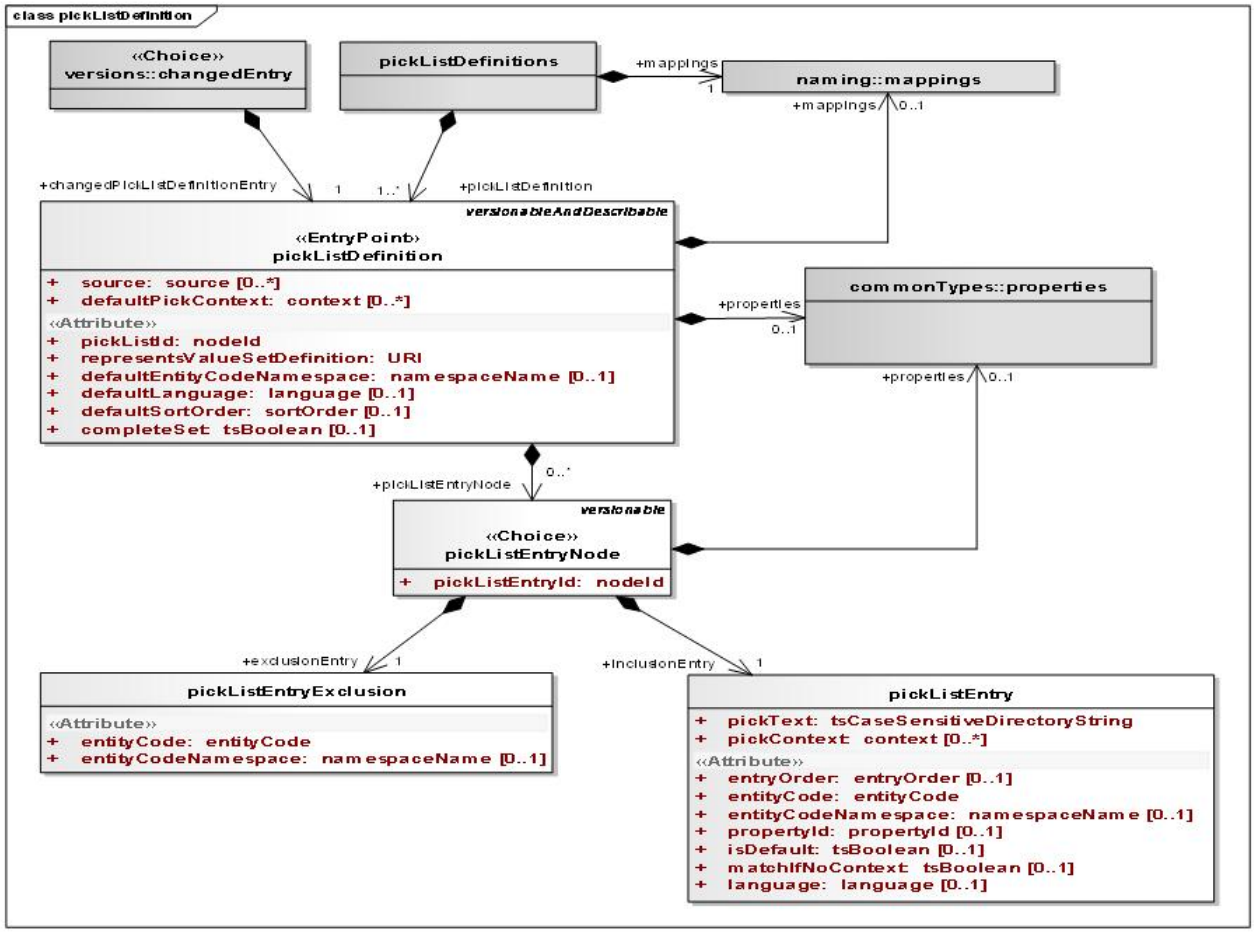
```
sh LoadValueSetDefinition.sh -in "file:///path/to/file.xml"
```

Pick List Definition

An ordered list of entity codes and corresponding presentations drawn from a resolved value set definition.

Pick List Definition logical model

Here is a UML representation of a Pick List Definition with in LexGrid 201001 model:



Pick List Definition logical model description

Pick List Definition

An ordered list of entity codes and corresponding presentations drawn from a resolved value set definition.

Attributes of Pick List Definition :
Source: The local identifiers of the source(s) of this pick list definition. Must match a local id of a supportedSource in the corresponding mappings section.
pickListId: An identifier that uniquely names this list within the context of the collection.
representsValueSetDefinition: The URI of the value set definition that is represented by this pick list
defaultEntityCodeNamespace: Local name of the namespace to which the entry codes in this list belong. defaultEntityCodeNamespace must match a local id of a supportedNamespace in the mappings section.
defaultLanguage: The local identifier of the language that is used to generate the text of this pick list if not otherwise specified. Note that this language does NOT necessarily have any coorelation with the language of a pickListEntry itself or the language of the target user. defaultLanguage must match a local id of a supportedLanguage in the mappings section.
defaultSortOrder: The local identifier of a sort order that is used as the default in the definition of the pick list
defaultPickContext: The local identifiers of the context used in the definition of the pick list.
completeSet: True means that this pick list should represent all of the entries in the resolved value set definition. Any active entity codes that aren't in the specific pick list entries are added to the end, using the designations identified by the defaultLanguage, defaultSortOrder and defaultPickContext. Default: false

Pick List Entry Node

An inclusion (pickListEntry) or exclusion (pickListEntryExclusion) in a pick list definition

Attributes of Pick List Entry Node :
pickListEntryId: Unique identifier of this node within the list.

Pick List Entry

An entity code and corresponding textual representation.

Attributes of Pick List Entry :
pickText: The text that represents this node in the pick list. Some business rules may require that this string match a presentation associated with the entityCode
pickContext: The local identifiers of the context(s) in which this entry applies. pickContext must match a local id of a supportedContext in the mappings section
entryOrder: Relative order of this entry in the list. pickListEntries without a supplied order follow the all entries with an order, and the order is not defined.
entityCode: Entity code associated with this entry.
entityCodeNamespace: Local identifier of the namespace of the entity code if different than the pickListDefinition defaultEntityCodeNamespace. entityCodeNamespace must match a local id of a supportedNamespace in the mappings section.
propertyId: The property identifier associated with the entityCode and entityCodeNamespace that the pickText was derived from. If absent, the pick text can be anything. Some terminologies may have business rules requiring this attribute to be present.
isDefault: True means that this is the default entry for the supplied language and context.
matchIfNoContext: True means that this entry can be used if no contexts are supplied, even though pickContext ispresent.
Language: The local name of the language to be used when the application/user supplies a selection language matches. If absent, this matches all languages. language must match a local id od of a supportedLanguage in the mappings section.

Pick List Entry Exclusion

An entity code that is explicitly excluded from a pick list.

Attributes of Pick List Entry Exclusion:
entityCode: Entity code associated with this entry.
entityCodeNamespace: Local identifier of the namespace of the entity code if different than the pickListDefinition defaultEntityCodeNamespace. entityCodeNamespace must match a local id of a supportedNamespace in the mappings section.

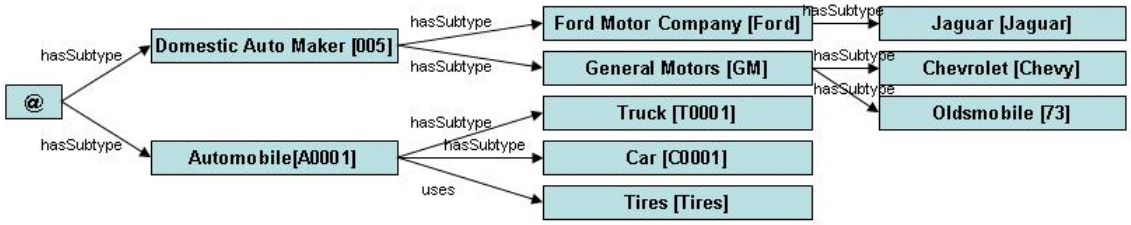
Possible forms of Pick List Definitions

- Ability to include all the concept codes contained in the referenced value set resolution by setting completeSet flag to 'true'
- Ability to include individual pickText derived from concept code designation of the referenced code system
- Ability to exclude concept codes from the pick list
- Ability to combine of any of the above

Fictitious examples of Pick List Definitions

Example 1

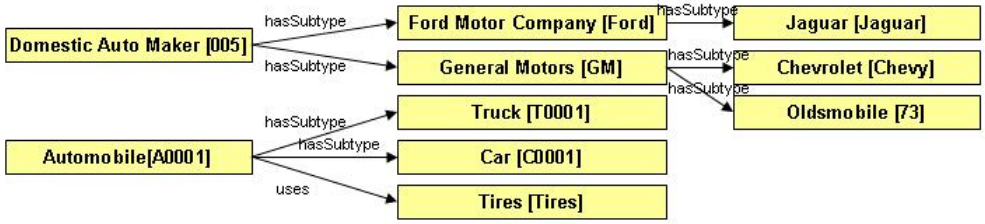
Say, we have a Code System 'Automobiles' that contains following concept codes and we also have a Value Set Definition 'urn:VDExample1' that contains all the concepts (except root code '@')from this code system:



So to define pick list definition to include all the preferred designations from the concept codes that are contained in Resolved Value Set Definition 'urn:VDExample1', we could define it as simple as:

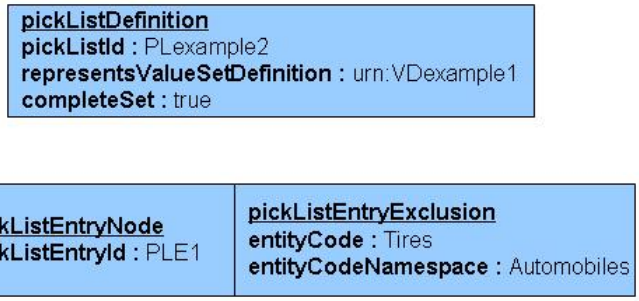
pickListDefinition
pickListId : PLexample1
representsValueSetDefinition : urn:VDexample1
completeSet : true

And when this definition is resolved, we will get back all the preferred designations from the following concept codes :

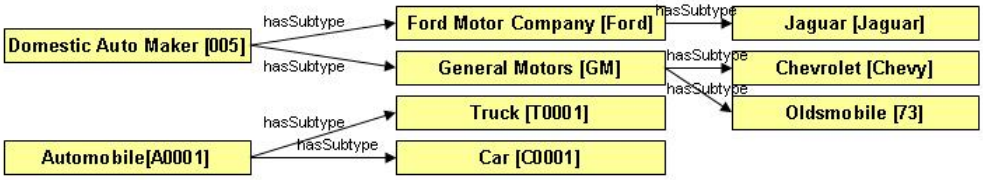


Example 2

Similar to above example, using the same Value Set Definition 'urn:VDExample1', but this time we want to exclude concept code 'Tires'. This could be done by introducing pickListEntryNode for exclusion as below:



And when this definition is resolved, we will get back all the preferred designations from the following concept codes :



Pick List Definition Versions

Pick List Definitions are versioned. The version of Pick List Definition changes when ever the definition is changed, it could be changing, adding or removing pickListEntryNode.

Pick List Services

LexEVS Pick List Services is an integral part of LexEVS Core API. It provides:

- Ability to load Pick List Definitions into LexGrid repository
- Ability to apply user restrictions and dynamically resolve the definitions during run time
- Ability to author Pick List Definition

Major functions provided by Pick List Services

- **Administration functions:**
  - Ability to load Pick List Definitions into LexGrid repository.
  - Ability to remove Pick List Definitions from LexGrid repository.
  - **Here are some of the admin methods available:**
    - **loadPickList(PickListDefinition pldef, String systemReleaseURI)** - Loads supplied Pick List Definition object
    - **loadPickList(InputStream inputStream, boolean failOnAllErrors)** - Loads Pick List Definitions found in inputStream
    - **loadPickList (String xmlFileLocation, boolean failOnAllErrors)** - Loads Pick List Definitions found in input xml file
    - **removePickList(String pickListId)** - Removes supplied Pick List Definition from the system

- **Query functions:**
  - Ability to list all the Pick List Definitions loaded in the system.
  - Ability to dynamically resolve Pick List Definition with/without user supplied restrictions.
  - Ability to list all the Pick List Definitions that uses given Value Set Definition URI.
  - **Here are some of the query methods available:**
    - **getPickListDefinitionById(String pickListId)** - Returns pickList definition for supplied pickListId
    - **getPickListDefinitionsForValueSetDefinition(URI valueSetDefinitionURI)** - Returns all the pickList definitions that represents supplied value set definition URI

        - **getPickListValueSetDefinition(String pickListId)** - Returns an URI of the represented value set definition of the pickList
        - **listPickListIds()** - Returns a list of pickListIds that are available in the system
        - **resolvePickList(String pickListId, boolean sortByText)** - Resolves pickList definition for supplied pickListId
        - **resolvePickListForTerm(String pickListId, String term, String matchAlgorithm, String language, String[] context, boolean sortByText)** - Resolves pickList definition by applying supplied arguments
            - **ResolvedPickListEntryList :** Contains the list of resolved Pick List Entries. Also provides helpful features to add, remove, enumerate Pick List Entries.

- **Authoring functions:**
    - **pickListDefinition** and **pickListEntryNode** classes are versionable entities in LexGrid model. Following authoring functions can be performed on them:
        - **NEW :** Create new Pick List Definition or add new pickListEntryNode to existing Pick List Definition.
        - **MODIFY :** Modify existing attributes of Pick List Definition or pickListEntryNode.
        - **REMOVE :** Ability to completely remove Pick List Definition or pickListEntryNode. This is not the same as deprecated, as the entity ceases to exist in future versions.

Refer to Revision section of this document for information on how revisions can be applied and how LexEVS handles them.

CTS 2 specific Pick List Services

There are **NO** functional specifications specific to Pick List in CTS 2 SFM.

Load Scripts

Scripts to load Pick List Definitions into LexEVS system will be located under 'Admin' folder of LexEVS install directory. This loader scripts will only load data in XML file that is in LexGrid format.

Pick List Loader

LoadPickList.bat for Windows environment and LoadPickList.sh for Unix environment.
Both these scripts takes in following parameters:

Parameter	Function
-in	Input <uri> URI or path specifying location of the source file.
-v	Validate <int> Perform validation of the candidate resource without loading data. Supported levels of validation includes : 0 = Verify document is well-formed 1 = Verify document is valid

Example:

```
sh LoadPickList.sh -in "file:///path/to/file.xml"
```