- Home
- Knowledge Centers
  - caGrid
  - Clinical Trials Management Systems
  - Data Sharing and Intellectual Capital
  - Molecular Analysis Tools
  - Tissue/Biospecimen Banking and Technology Tool
  - Vocabulary
- Discussion Forums
  - caBIG General Forum
  - caGrid
  - Clinical Trials Management Systems
  - Data Sharing and Intellectual Capital
  - Molecular Analysis Tools
  - Tissue/Biospecimen Banking and Technology Tool
  - Vocabulary
- Bugs/Feature Requests
- Development Code Repository

LexEVS 5.x Loader Guide

From Vocab_Wiki

LexEVS 5.x Loader Guide > Main Page > LexEVS 5.x Included Loaders > Main Page > LexEVS 5.x Loader Guide

Introduction

This guide is intended for the LexEVS developer. It provides information about the loaders provided, mapping, and how to create your own loaders using the loader framework. It contains the following sections:

1. Included Loaders
2. Loader Model Elements Mapping
3. Loader Source Mapping
4. Loader Framework

Related documents

- Installation Guide for information about software requirements and configuring your environment
- Programmer's Guide for information about using the LexEVS core services and APIs

Retrieved from "https://cabig-kc.nci.nih.gov/Vocab/KC/index.php/LexEVS_5.x_Loader_Guide"
Categories: VKC Contents | Documentation | LexEVS

- This page was last modified on 22 December 2009, at 13:00.

CONTACT USPRIVACY NOTICEDISCLAIMERACCESSIBILITYAPPLICATION SUPPORT

National Cancer Institute U.S. National Institutes of Health | www.cancer.gov

caBIG Knowledge Center
A part of the Enterprise Support Network

- Home
- Knowledge Centers
  - caGrid
  - Clinical Trials Management Systems
  - Data Sharing and Intellectual Capital
  - Molecular Analysis Tools
  - Tissue/Biospecimen Banking and Technology Tool
  - Vocabulary
- Discussion Forums
  - caBIG General Forum
  - caGrid
  - Clinical Trials Management Systems
  - Data Sharing and Intellectual Capital
  - Molecular Analysis Tools
  - Tissue/Biospecimen Banking and Technology Tool
  - Vocabulary
- Bugs/Feature Requests
- Development Code Repository

LexEVS 5.x Included Loaders

From Vocab_Wiki

LexEVS 5.x Loader Framework > Main Page > LexEVS 5.x Loader Guide > Main Page > LexEVS 5.x Included Loaders

Contents

Introduction

This document is a section of the Loader Guide. It was formerly the LexEVS 5.0 *Supported Loaders* guide.

LexEVS 5.x includes the loaders listed in this document. You can also create your own loaders using the

Loader Framework extension.

NCI MetaThesaurus Loader

Validates and/or loads the complete NCI MetaThesaurus. Content is supplied in RRF format. Note: To load individual coding schemes, consider using the UMLS_Loader as an alternative.

OBO Loader

Validates and/or loads content provided in Open Biomedical Ontologies (OBO) text format.

OWL Loader

Validates and/or loads content provided in Web Ontology Language (OWL) XML format. Note that for LexEVS phase 1 this loader is designed to specifically handle the NCI Thesaurus as provided in OWL format.

Text Loader

A loader for delimited text type files. Text files come in one of two formats: indented code/designation pair or indented code/designation/description triples.

UMLS Loader

Load one or more coding schemes from UMLS RRF format stored in a SQL database.

MetaData Loader

Validates and/or loads content provided in metadata xml format. The only requirement of the xml file is that it be a valid xml file.

NCI History Loader

A loader that takes the delimited NCI history file and applies it to a coding scheme.

OBO History Loader

Load an OBO change history file.

Retrieved from "https://cabig-kc.nci.nih.gov/Vocab/KC/index.php/LexEVS_5.x_Included_Loaders"
Categories: VKC Contents | Documentation | LexEVS

- This page was last modified on 6 December 2009, at 05:04.

CONTACT US PRIVACY NOTICE DISCLAIMER ACCESSIBILITY APPLICATION SUPPORT

National Cancer Institute

caBIG Knowledge Center
A part of the Enterprise Support Network

U.S. National Institutes of Health | www.cancer.gov

- Home
- Knowledge Centers
  - caGrid
  - Clinical Trials Management Systems
  - Data Sharing and Intellectual Capital
  - Molecular Analysis Tools
  - Tissue/Biospecimen Banking and Technology Tool
  - Vocabulary
- Discussion Forums
  - caBIG General Forum
  - caGrid
  - Clinical Trials Management Systems
  - Data Sharing and Intellectual Capital
  - Molecular Analysis Tools
  - Tissue/Biospecimen Banking and Technology Tool
  - Vocabulary
- Bugs/Feature Requests
- Development Code Repository

LexEVS 5.x Loader Model Elements Mapping

From Vocab_Wiki

LexEVS Version 5.0 FAQ > Main Page > LexEVS 5.x Loader Guide > Main Page > LexEVS 5.x Loader Model Elements Mapping

Contents

Introduction

This document is a section of the Loader Guide. It was formerly the LexEVS v5.0 *Loader Mapping Guide*.

For the LexEVS v5.1 enhancements to the RRF loader, see the Loader Source Mapping section.

OWL Mapping - 4.2.1

OWL: RDF Schema Features

OWL Mapping - Protégé (4.2.1)		

OWL Element	LexGrid	Comments
OWL: RDF Schema Features		
owl:ontology	codingScheme	
xml:lang	codingScheme.defaultLanguage	Default is 'en'
dc:title	codingScheme.formalName	
rdfs:label	codingScheme.localName	
URI	codingScheme.registeredName	
owl:versionInfo	codingScheme.representsVersion	Default is 'UNASSIGNED'
dc:rights	codingScheme.copyright	
owl:Class (Thing, Nothing)	concept	
rdf:ID	concept.conceptCode	
	concept.isActive	Hard coded as "Active"
	concept.isAnonymous	
rsfs:label	concept.entityDescription	
rdf:comment	concept.comment	
rdfs:subClassOf	association	
	association.id = "subClassOf"	
	association.forwardName = "subClassOf"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="false"	
	association.isTransitive="true"	
rdf:Property (ObjectProperty)	association	An association between two classes (hasDomain, hasRange)
	association concept.conceptProperty	An association between one class (domain) and one asscoication (hasDomain and hasDataProperty). The conceptProperty defines the range.
rdfs:subPropertyOf	association	
	association.id = "subPropertyOf"	
	association.forwardName = "subPropertyOf"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="false"	
	association.isTransitive="true"	
rdfs:domain	association	
	association.id = "hasDomain"	
	association.forwardName = "hasDomain"	
	association.isNavigable = "true"	
	association.isReflexive="false"	
	association.isSymmetric="false"	
	association.isTransitive="true"	
rdfs:range	association	
	association.id = "hasRange"	
	association.forwardName = "hasRange"	
	association.isNavigable = "true"	
	association.isReflexive="false"	
	association.isSymmetric="false"	
	association.isTransitive="false"	
Individual	association	A 'hasInstance' association is created. (ie. sourceId = Country, targetId = America)
	association.id = "hasInstance"	

OWL: (In)Equality

OWL Mapping - Protégé (4.2.1)		
OWL Element	**LexGrid**	**Comments**
OWL: (In)Equality		
owl:equivalentClass	association	
	association.id = "equivalentClass"	
	association.forwardName = "equivalentClass"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	

	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName="equivalentClass"	
owl:equivalentProperty	association	
	association.id = "equivalentProperty"	
	association.forwardName = "equivalentProperty"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName="equivalentProperty"	
owl:sameAs	association	
	association.id = "sameAs"	
	association.forwardName = "sameAs"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName="sameAs"	
differentFrom	association	
	association.id = "differentFrom"	
	association.forwardName = "differentFrom"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName= "differentFrom"	
owl:AllDifferent	association	
	association.id = "AllDifferent"	
	association.forwardName = "AllDifferent"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName= "AllDifferent"	

OWL: Property Characteristics

OWL Mapping - Protégé (4.2.1)		
OWL Element	**LexGrid**	**Comments**
OWL: Property Characteristics		
owl:inverseOf	association	
	association.id = "inverseOf"	
	association.forwardName = "inverseOf"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName="inverseOf"	
owl:TransitiveProperty	association.isTransitive	association property 'isTransitive'
owl:SymmetricProperty	association.isSymmetric	association property 'isSymmetric'
owl:InverseFunctionalProperty	association.isReverseFunctional	association property 'isReverseFunctional'
owl:FunctionalProperty	association.isFunctional	association property 'isFunctional'

OWL: Property Restrictions

OWL Mapping - Protégé (4.2.1)		
OWL Element	**LexGrid**	**Comments**
OWL: Property Restrictions		

owl:Restriction	concept	Create an anonymous concept for the restriction
	concept.id	System generated
	concept.isActive = true	
	concept.isAnonymous = true	Hardcoded "True"
owl:onProperty	association.id	
owl: allValuesFrom	concept.entityDescription	String of allValuesFrom values
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of allValuesFrom values
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:unionOf"	
owl: someValuesFrom	concept.entityDescription	String of someValuesFrom values
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of someValuesFrom values
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:intersectionOf"	
owl:intersectionOf	concept.entityDescription	String of intersectionOf values (ie. Pizza and not VegetarianPizza)
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of intersectionOf values (ie. Pizza and not VegetarianPizza)
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:intersectionOf"	
UnionOf	concept.conceptProperty.text = "owl:unionOf"	
owl:complementOf	association	association.id = "subClassOf"
owl:oneOf	concept.entityDescription	String of oneOf values
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of oneOf values
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:intersectionOf"	
owl:hasValue	associationQualification.nameAndValueList.content	
owl:minCardinality	concept.entityDescription	String of minCardinality Values (ie. (hasTopping min 3) and Pizza)
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of minCardinality Value (ie. (hasTopping min 3) and Pizza)
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:intersectionOf"	
owl:maxCardinality	concept.entityDescription	String of maxCardinality Values (ie. (hasTopping max 2) and Pizza)
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of maxCardinality Values (ie. (hasTopping max 2) and Pizza)
		Generated value for property using "P" concatenated with a steadily incremented

	concept.conceptProperty.propertyId	numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:intersectionOf"	
		String of cardinality Values
owl:cardinality	concept.entityDescription	
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of cardinality Values
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:intersectionOf"	
owl:disjointWith	association	association.id = "disjointWith"

OWL: Annotation Property

OWL Mapping - Protégé (4.2.1)		
OWL Element	**LexGrid**	**Comments**
OWL: Annotation Property		
rdfs:label	Presentation	
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName = "textualPresentation"	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	Value of rdfs:label
rdfs:comment	Comment	
	concept.comment.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.comment.propertyName = "comment"	Hardcoded "comment"
	concept.presentation.text	Value of rdfs:comment
rdfs:seeAlso	conceptProperty	
rdfs:isDefinedBy	conceptProperty	

OWL: Versioning

OWL Mapping - Protégé (4.2.1)		
OWL Element	**LexGrid**	**Comments**
OWL: Versioning		
owl:versionInfo	codingScheme.representsVersion	
priorVersion		Not Mapped
backwardCompatibleWith		Not Mapped
owl:incompatibleWith	association	
	association.id = "incompatibleWith"	
	association.forwardName = "incompatibleWith"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName="incompatibleWith"	
DeprecatedClass	Concept attribute setIsActive = false	Not Mapped
DeprecatedProperty		Not Mapped

OWL Mapping - 5.0

OWL: RDF Schema Features

OWL Mapping - Protégé (5.0)		
OWL Element	**LexEVS**	**Comments**
OWL: RDF Schema Features		
owl:ontology	codingScheme	
xml:lang	codingScheme.defaultLanguage	Default is 'en'

dc:title	codingScheme.formalName	
rdfs:label	codingScheme.localName	
URI	codingScheme.registeredName	
owl:versionInfo	codingScheme.representsVersion	Default is 'UNASSIGNED'
dc:rights	codingScheme.copyright	
owl:Class (Thing, Nothing)	concept	
rdf:ID	concept.conceptCode	
	concept.isActive	Hard coded as "Active"
	concept.isAnonymous	
	concept.isDefined	
rsfs:label	concept.entityDescription	
rdf:comment	concept.comment	
rdfs:subClassOf	association	
	association.id = "subClassOf"	
	association.forwardName = "subClassOf"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="false"	
	association.isTransitive="true"	
rdf:Property (ObjectProperty)	association	An association between two classes (domain, range)
	association concept.conceptProperty	An association between one class (domain) and one asscoication (domain and hasDataProperty). The conceptProperty defines the range.
rdfs:subPropertyOf	association	
	association.id = "subPropertyOf"	
	association.forwardName = "subPropertyOf"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="false"	
	association.isTransitive="true"	
rdfs:domain	association	
	association.id = "domain"	
	association.forwardName = "domain"	
	association.isNavigable = "true"	
	association.isReflexive="false"	
	association.isSymmetric="false"	
	association.isTransitive="true"	
rdfs:range	association	
	association.id = "range"	
	association.forwardName = "range"	
	association.isNavigable = "true"	
	association.isReflexive="false"	
	association.isSymmetric="false"	
	association.isTransitive="false"	
Individual	association	An 'instance' association is created. (ie. sourceId = Country, targetId = America)
	association.id = "instance"	

OWL: (In)Equality

OWL Mapping - Protégé (5.0)		
OWL Element	**LexEVS**	**Comments**
OWL: (In)Equality		
owl:equivalentClass	association	
	association.id = "equivalentClass"	
	association.forwardName = "equivalentClass"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName="equivalentClass"	
owl:equivalentProperty	association	

	association.id = "equivalentProperty"	
	association.forwardName = "equivalentProperty"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName="equivalentProperty"	
owl:sameAs	association	
	association.id = "sameAs"	
	association.forwardName = "sameAs"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName="sameAs"	
differentFrom	association	
	association.id = "differentFrom"	
	association.forwardName = "differentFrom"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName= "differentFrom"	
owl:AllDifferent	association	
	association.id = "AllDifferent"	
	association.forwardName = "AllDifferent"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName= "AllDifferent"	

OWL: Property Characteristics

OWL Mapping - Protégé (5.0)		
OWL Element	**LexEVS**	**Comments**
OWL: Property Characteristics		
owl:inverseOf	association	
	association.id = "inverseOf"	
	association.forwardName = "inverseOf"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName="inverseOf"	
owl:TransitiveProperty	association.isTransitive	association property 'isTransitive'
owl:SymmetricProperty	association.isSymmetric	association property 'isSymmetric'
owl:InverseFunctionalProperty	association.isReverseFunctional	association property 'isReverseFunctional'
owl:FunctionalProperty	association.isFunctional	association property 'isFunctional'

OWL: Property Restrictions

OWL Mapping - Protégé (5.0)		
OWL Element	**LexEVS**	**Comments**
OWL: Property Restrictions		
owl:Restriction	concept	Create an anonymous concept for the restriction
	concept.id	System generated
	concept.isActive = true	
	concept.isAnonymous = true	Hardcoded "True"
owl:onProperty	association.id	

owl: allValuesFrom	concept.entityDescription	String of allValuesFrom values
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of allValuesFrom values
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:unionOf"	
owl: someValuesFrom	concept.entityDescription	String of someValuesFrom values
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of someValuesFrom values
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:intersectionOf"	
owl:intersectionOf	concept.entityDescription	String of intersectionOf values (ie. Pizza and not VegetarianPizza)
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of intersectionOf values (ie. Pizza and not VegetarianPizza)
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:intersectionOf"	
UnionOf	concept.conceptProperty.text = "owl:unionOf"	
owl:complementOf	association	association.id = "subClassOf"
owl:oneOf	concept.entityDescription	String of oneOf values
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of oneOf values
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:intersectionOf"	
owl:hasValue	associationQualification.nameAndValueList.content	
owl:minCardinality	concept.entityDescription	String of minCardinality Values (ie. (hasTopping min 3) and Pizza)
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of minCardinality Value (ie. (hasTopping min 3) and Pizza)
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:intersectionOf"	
owl:maxCardinality	concept.entityDescription	String of maxCardinality Values (ie. (hasTopping max 2) and Pizza)
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of maxCardinality Values (ie. (hasTopping max 2) and Pizza)
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:intersectionOf"	
		String of cardinality Values

owl:cardinality	concept.entityDescription	
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	String of cardinality Values
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:intersectionOf"	

OWL: Annotation Property

OWL Mapping - Protégé (5.0)		
OWL Element	**LexEVS**	**Comments**
owl:disjointWith	association	association.id = "disjointWith"
OWL: Annotation Property		
rdfs:label	Presentation	
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName = "textualPresentation"	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	Value of rdfs:label
rdfs:comment	Comment	
	concept.comment.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.comment.propertyName = "comment"	Hardcoded "comment"
	concept.presentation.text	Value of rdfs:comment
rdfs:seeAlso	conceptProperty	
rdfs:isDefinedBy	conceptProperty	

OWL: Versioning

OWL Mapping - Protégé (5.0)		
OWL Element	**LexEVS**	**Comments**
OWL: Versioning		
owl:versionInfo	codingScheme.representsVersion	
priorVersion		Not Mapped
backwardCompatibleWith		Not Mapped
owl:incompatibleWith	association	
	association.id = "incompatibleWith"	
	association.forwardName = "incompatibleWith"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName="incompatibleWith"	
DeprecatedClass	Concept attribute setIsActive = false	Not Mapped
DeprecatedProperty		Not Mapped

OWL Mapping - NCI OWL

OWL: RDF Schema Features

OWL Mapping - NCI OWL		
OWL Element	**LexGrid**	**Comments**
OWL: RDF Schema Features		
owl:ontology	codingScheme	Hardcoded "NCI_Thesaurus"
xml:lang	codingScheme.defaultLanguage	Hardcoded "en"
dc:title	codingScheme.formalName	Hardcoded "NCI Thesaurus"
rdfs:label	codingScheme.localName	Hardcoded "NCI_Thesaurus"
		Hardcoded "40010"
		Hardcoded "urn:oid:2.16.840.1.113883.3.26.1.1"
URI	codingScheme.registeredName	Hardcoded "http://ncicb.nci.nih.gov/xml/owl/EVS/Thesaurus.owl#"

owl:versionInfo	codingScheme.representsVersion	
dc:rights	codingScheme.copyright	Read from hardcoded "Terms.txt" file .
rdfs:comment	codingScheme.entityDescription	
	codingScheme.isNative	Hardcoded "true"
owl:Class (Thing, Nothing)	concept	
code	concept.id	
	concept.isActive	Hard coded as "true" unless class "owl:DeprecatedClass", then 'false'
	concept.isAnonymous	
rsfs:label	concept.entityDescription	
rdf:comment	concept.comment	
	conceptProperty	Indicate whether the concept is primative (has no equavalent classes)
	concept.conceptProperty.propertyName	Hard coded as "primitive"
	concept.conceptProperty.text	"true"
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	presentation	Provide default presentation to match concept entity description if not provided as property
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "NCI_Preferred_Term"
rdfs:label	concept.presentation.text	concept.entityDescription
	conceptProperty	Property with designated concept name label (per NCI requirements and used in codeToName/nameToCode lookup).
	concept.conceptProperty.propertyName	Hard coded as "CONCEPT_NAME"
rdfs:label	concept.conceptProperty.text	concept.entityDescription
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	relation	Top-level container for associations (non-inheritable, non-defining relationships between concepts.
	relations.dc	Hard coded as "associations"
	relations.isNative	Hard coded as "true"
	relations.entityDescription	Hard coded as "Non-inheritable non-defining relations."
	relation	Top-level container for roles (inheritable relationships)
	relations.dc	Hard coded as "roles"
	relations.isNative	Hard coded as "true"
	relations.entityDescription	Hard coded as "Inheritable/defining relations."
rdfs:subClassOf	association	Association for subtype hierarchy.
	association.id = "hasSubtype"	
	association.forwardName = "hasSubtype"	
	association.reverseName = "isA"	
	association.isNavigable = "true"	Hard coded as "true"
	association.isReflexive="true"	Hard coded as "true"
	association.isSymmetric="false"	Hard coded as "false"
	association.isTransitive="true"	Hard coded as "true"
hasElement	association	Association used to register component classes as elements of anonymous node representations.
	association.id = "hasElement"	
	association.forwardName = "hasElement"	
	association.isNavigable = "true"	Hard coded as "true"
	association.isSymmetric="false"	Hard coded as "false"
	association.isTransitive="true"	Hard coded as "true"
rdfs:domain	association	Association for role_has_domain relations
	association.id = "Role_Has_Domain"	
	association.forwardName = "roleHasDomain"	
	association.reverseName = "kindIsDomainOf"	
	association.isNavigable = "true"	Hard coded as "true"
	association.isReflexive="false"	Hard coded as "false"
	association.isSymmetric="false"	Hard coded as "false"
	association.isTransitive="true"	Hard coded as "true"
rdfs:range	association	Association for range relations
	association.id = "Role_Has_Range"	
	association.forwardName = "roleHasRange"	
	association.reverseName = "kindIsRangeOf"	
	association.isNavigable = "true"	Hard coded as "true"
	association.isReflexive="false"	Hard coded as "false"

	association.isSymmetric="false"	Hard coded as "false"
	association.isTransitive="false"	Hard coded as "false"
rdf:Property (ObjectProperty)	association	An association between two classes (hasDomain, hasRange)
rdfs:subPropertyOf		Not Mapped

OWL: (In)Equality

OWL Mapping - NCI OWL		
OWL Element	**LexGrid**	**Comments**
OWL: (In)Equality		
owl:equivalentClass	association	Association for equivalent class.
	association.id = "equivalentClass"	
	association.forwardName = "equivalentClass"	
	association.reverseName = "equivalentClass"	
	association.isNavigable = "true"	Hard coded as "true"
	association.isReflexive="true"	Hard coded as "true"
	association.isSymmetric="true"	Hard coded as "true"
	association.isTransitive="true"	Hard coded as "true"

OWL: Property Characteristics

OWL Mapping - NCI OWL		
OWL Element	**LexGrid**	**Comments**
OWL: Property Characteristics		
owl:inverseOf	association	
	association.id = "inverseOf"	
	association.forwardName = "inverseOf"	
	association.isFunctional = "false"	
	association.isNavigable = "true"	
	association.isReflexive="true"	
	association.isSymmetric="true"	
	association.isTransitive="true"	
	association.reverseName="inverseOf"	
owl:TransitiveProperty	association.isTransitive	association property 'isTransitive'
owl:SymmetricProperty	association.isSymmetric	association property 'isSymmetric'
owl:InverseFunctionalProperty	association.isReverseFunctional	association property 'isReverseFunctional'
owl:FunctionalProperty	association.isFunctional	association property 'isFunctional'

OWL: Property Restrictions

OWL Mapping - NCI OWL		
OWL Element	**LexGrid**	**Comments**
OWL: Property Restrictions		
owl:Restriction	concept	Anonymous concept created.
	concept.entityDescription = "RestrictionOn: " + association name	Concatination of "Restriction On: " and assocation name
	concept.isAnonymous = true	
owl: allValuesFrom	associationQualification.association.Qualifier = "AllValuesFrom"	
owl: someValuesFrom	associationQualification.association.Qualifier = "someValuesFrom"	
owl:intersectionOf	concept.entityDescription	Concatination of "Restriction On: " and assocation name
	concept.isAnonymous = true	
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	Set to concept.entityDescription
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:intersectionOf"	
owl:unionOf	concept.entityDescription	Concatination of "Restriction On: " and assocation name
	concept.isAnonymous = true	
	concept.presentation.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.

	concept.presentation.propertyName	Hardcoded "textualPresentation"
	concept.presentation.isPreferred = true	Hardcoded "true"
	concept.presentation.text	Set to concept.entityDescription
	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = type	Hardcoded "type"
	concept.conceptProperty.text = "owl:unionOf"	
owl:oneOf	concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
	concept.conceptProperty.propertyName = "owl:oneOf"	Hardcoded "owl:oneOf"
	concept.conceptProperty.text	String of oneOf values

OWL: Annotation Property

OWL Mapping - NCI OWL		
OWL Element	**LexGrid**	**Comments**
OWL: Annotation Property		
rdfs:comment	Comment	
	concept.comment.propertyId	Generated value for property textual presentation using "P" concatenated with a steadily incremented numerical value.
	concept.comment.propertyName = "comment"	Hardcoded "comment"
	concept.presentation.text	Value of rdfs:comment
rdfs:seeAlso	conceptProperty	
rdfs:isDefinedBy	conceptProperty	

OWL: Versioning

OWL Mapping - NCI OWL		
OWL Element	**LexGrid**	**Comments**
OWL: Versioning		
owl:versionInfo	codingScheme.representsVersion	
priorVersion		Not Mapped
backwardCompatibleWith		Not Mapped
DeprecatedClass		Not Mapped
DeprecatedProperty		Not Mapped

Legacy Complex Properties Mapping

Legacy Complex Properties Mapping							
tag	presentation	source	represenational form	qualifier	model element	value column name	model element
go-term	x					propertyValue	
go-id				x	propertyQualifierId	val1	PropertyQualifier attribute content?
go-source				x	propertyQualifierId	val1	PropertyQualifier attribute content?
source-date				x	propertyQualifierId	val1	PropertyQualifier attribute content?
term-name	x					propertyValue	
term-group			x			representationalForm	property attribute
term-source		x				attributeValue	source
def-source		x				attributeValue	source
def-definition	x					propertyValue	definition
Definition_Review_Date				x	propertyQualifierId	val1	PropertyQualifier attribute content?
Definition_Reviewer_Name				x	propertyQualifierId	val1	PropertyQualifier attribute content?

UMLS SemNet Mapping

Coding Scheme

UMLS SemNet Mapping					
RRF File Name	***RRF Column Name***	***RRF Definition***	***NCI Meta only***	***LexGrid Model Element***	***comments***
Coding Scheme					
				codingScheme.representsVersion	
				codingScheme.codingScheme	hard coded in java file as "UMLS_SemNet"
				codingScheme.formalName	hard coded in java file as "UMLS Semantic Network"
				codingScheme.defaultLanguage	hard coded in java file as "en"
				codingScheme.approxNumConcepts	hard coded in java file as

				codingScheme.entityDescription	hard coded in java file as "The UMLS Semantic Network is one of three UMLS Knowledge Sources developed as part of the Unified Medical Language System project. The network provides a consistent categorization of all concepts represented in the UMLS Metathesaurus."
license.txt				codingScheme.copyright	Read from license.txt file or hard coded reference in java file
				codingScheme.registeredName	hard coded in java file as "urn:lsid:nlm.nih.gov:semnet"
				codingScheme.concepts.dc	hard coded in java file as "concepts"
				codingScheme.relations.dc	hard coded in java file as "relations"
				codingScheme.mappings.dc	hard coded in java file as "mappings"
				codingScheme.localNameList	
				codingScheme.localNameList.	hard coded in java file as "UMLS_SemNet"
				codingScheme.localNameList	
				codingScheme.localNameList.	
				codingScheme.source	
				codingScheme.source.content	
				codingScheme.localNameList	
				codingScheme.localNameList.	
				codingScheme.localNameList	
				codingScheme.localNameList.	
				codingScheme.localNameList	
				codingScheme.localNameList.	
				codingScheme.localNameList	
				codingScheme.localNameList.	
				mappings.supportedFormat	
				mappings.supportedFormat.localId	hard coded in java file as "text/plain"
				mappings.supportedFormat.urn	hard coded in java file as "urn:oid:2.16.840.1.113883.6.10:text_plain"
				mappings.supportedAssociation	
SRDEF	RL			mappings.supportedAssociation.localId	
				mappings.supportedContext	
				mappings.supportedSource	
				mappings.supportedSource.localId	hard coded in java file as "NLM"
				mappings.supportedSource.urn	hard coded in java file as "urn:lsid:nlm.nih.gov"
				mappings.supportedHierarchy	
				mappings.supportedHierarchy.localId	hard coded in java file as "is_a"
				mappings.supportedHierarchy.isForwardNavigable	hard coded as "true"
				mappings.supportedHierarchy.rootCode	hard coded as "@"
				mappings.supportedHierarchy.associationList	hard coded in java file as "hasSubtype"
				mappings.supportedAssociationQualifier	
SRFLD	COL			mappings.supportedProperty	
				mappings.supportedProperty.localId	If SRDEF appears in the FIL column then this is treated a potential supported property and is entered in supported properties as such.
				mappings.supportedProperty.urn	hard coded in java file as ""
				mappings.supportedLanguage	
				mappings.supportedLanguage.localId	hard coded in java file as "en"
				mappings.supportedLanguage.urn	hard coded in java file as "urn:oid:2.16.840.1.113883.6.84:en"
				mappings.supportedCodingScheme	
				mappings.supportedCodingScheme.localId	hard coded in java file as "UMLS_SemNet"
				mappings.supportedCodingScheme.urn	hard coded in java file as "urn:lsid:nlm.nih.gov:semnet"
				mappings.supportedRepresentationalForm	
				mappings.supportedConceptStatus	
				mappings.supportedPropertyLink	
				mappings.supportedPropertyQualifier	
				mappings.supportedDataType	

Concepts

UMLS SemNet Mapping

RRF File Name	***RRF Column Name***	***RRF Definition***	***NCI Meta only***	***LexGrid Model Element***	***comments***
Concepts					
SRDEF	UI			concept.id(inherited from Entity)	
SRDEF	STY/RL			concept.enitityDescription(inheritance path Entity->versionableAndDescribable)	
				concept.conceptProperty	
SRDEF	NH			concept.conceptProperty.text.content	
				concept.conceptProperty.format	hard coded in java file as "text/plain"
				concept.conceptProperty.propertyName	hard coded in java file as "NH"

				concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
				concept.presentation	
				concept.presentation.propertyName (inherited from Property)	Hard coded in java file as "STY/RL" or "ABR"
				concept.presentation.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
SRDEF	STY/RL, ABR			concept.presentation.text.content	
				concept.presentation.format	hard coded in java file as "text/plain"
				concept.presentation.isPreferred	hard coded in java file as true.
				concept.definition.propertyName (inherited from Property)	Hard coded in java file as "DEF"
				concept.definition.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
SRDEF	DEF			concept.definition.text.content	
				concept.definition.format	hard coded in java file as "text/plain"
				concept.definition.isPreferred	hard coded in java file as true.
				concept.comment	
SRDEF	EX			concept.comment.propertyName (inherited from Property)	Hard coded in java file as "EX"
				concept.comment.text.content	
				concept.comment.format	hard coded in java file as "text/plain"
				concept.comment.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
				concept.instruction	
				concept.instruction.propertyName (inherited from Property)	Hard coded in java file as "UN"
SRDEF	UN			concept.instruction.text.content	
				concept.instruction.format	hard coded in java file as "text/plain"
				concept.instruction.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.

Relations

UMLS SemNet Mapping					
RRF File Name	***RRF Column Name***	***RRF Definition***	***NCI Meta only***	***LexGrid Model Element***	***comments***
Relations					
SRSTR	RL			association.id (inherited from Entity)	In the case of RL value is "isa" the id is hard coded to hasSubtype. The direction of the association is also reversed
				association.isTransitive	hard coded to true if the value of RL is "isa"
SRSTR	RL			association.forwardName	Reversed when value of RL is "isa"
SRSTR	STY/RL			associationInstance.sourceId	Reversed when value of RL is "isa"
SRSTR	STY/RL			associationTarget.targetId	
SRDEF	RIN			association.reverseName	
SRDEF	DEF			association.entityDescription.content (inheritance path for entityDescription is Entity->versionableAndDescribable)	When SRDEF value RT is "RL"
SRSTRE1	UI/STY(first argument)			associationInstance.sourceId	Reversed when value of RL is "isa"
SRSTRE1	UI/STY(2nd argument)			associationTarget.targetId	Reversed when value of RL is "isa"

UMLS Mapping

This section will be updated with the RRF loader enhancements implemented in LexEVS v5.1. Until then, go to the LexEVS 5.x Loader Source Mapping section of this guide.

Coding Scheme

UMLS Mapping					
RRF File Name	***RRF Column Name***	***RRF Definition***	***NCI Meta only***	***LexGrid Model Element***	***comments***
Coding Scheme					
MRSAB.RRF	SVER	Release date or version number of a source		codingScheme.representsVersion	
MRSAB.RRF	SSN	Source short name		codingScheme.codingScheme	
MRSAB.RRF	SON	Source Official Name		codingScheme.formalName	

MRSAB.RRF	LAT	Language of Term(s)		codingScheme.defaultLanguage	
MRSAB.RRF	TRF	Term frequency for a source		codingScheme.approxNumConcepts	
MRSAB.RRF	SCIT	Source citation		codingScheme.entityDescription	inherits entityDescription from versionableAndDescribable
MRSAB.RRF	SCC	Content contact info for a source		codingScheme.copyright	
				codingScheme.registeredName	Pulled from iso mapping configuration file using method getISOString (RSAB from MRSAB.RRF)
MRDOC.RRF	EXPL	Detailed explanation	x	codingScheme.representsVersion	Where Dockey = "RELEASE" and value = "umls.release.name"
			x	codingScheme.codingScheme	Hard coded in java file as "NCI MetaThesaurus"
			x	codingScheme.formalName	Hard coded in java file as "NCI MetaThesaurus"
			x	codingScheme.defaultLanguage	Hard coded in java file as "ENG"
MRCONSO.RRF			x	codingScheme.approxNumConcepts	Count of CODE value in MRCONSO.RRF
			x	codingScheme.entityDescription	Hard coded in java file as "NCI MetaThesaurus loaded from RRF files."
			x	codingScheme.copyright	Hard coded in java file as "Some material in the NCI Metathesaurus is from copyrighted sources of the respective copyright claimants. All sources appearing in the NCI Metathesaurus are licensed or authorized for NCI use. Users of the NCI Metathesaurus are responsible for compliance with the terms of these licenses and with any copyright restrictions and are referred to NCI Center of Bioinformatics for license terms and to the copyright notices appearing in the original sources, all of which are obtainable online by reference at http://ncimeta.nci.nih.gov/."
				codingScheme.localNameList	Hard coded as constant in java file as "localName"
MRSAB.RRF	SON	Source Official Name		codingScheme.localNameList.	
				codingScheme.localNameList	Hard coded as constant in java file as "localName"
				codingScheme.localNameList.	Pulled from iso mapping configuration file using method getISOString (RSAB from MRSAB.RRF)
				codingScheme.source	Hard coded as constant in java file as "source"
MRDOC.RRF	EXPL	Detailed explanation		codingScheme.source.content	String concatenation of "UMLS-" and value of EXPL
			x	codingScheme.localNameList	Hard coded as constant in java file as "localName"
			x	codingScheme.localNameList.	Hard coded in java file as "NCI Thesaurus"
			x	codingScheme.localNameList	Hard coded as constant in java file as "localName"
			x	codingScheme.localNameList.	Hard coded in java file as "NCI_Thesaurus"
			x	codingScheme.localNameList	Hard coded as constant in java file as "localName"
			x	codingScheme.localNameList.	Hard coded in java file as "10001"
			x	codingScheme.localNameList	Hard coded as constant in java file as "source"
			x	codingScheme.localNameList.	Hard coded in java file as "RRF Files"
				mappings.supportedFormat	Hard coded as constant in java file as "Format"
				mappings.supportedFormat.localId	Hard coded as one of several constants in a java file
				mappings.supportedAssociation	Hard coded as constant in java file as "Association"
MRREL.RRF	REL, RELA	Relationship, Relationship attribute		mappings.supportedAssociation.localId	
				mappings.supportedContext	Hard coded as constant in java file as "Context" May not be used in individual RRF load
				mappings.supportedSource	Hard coded as constant in java file as "Source" May not be used in individual RRF load
				mappings.supportedHierarchy	Hard coded as constant in java file as "Hierarchy"
				mappings.supportedAssociationQualifier	Hard coded as constant in java file as "AssociationQualifier"
				mappings.supportedProperty	Hard coded as constant in java file as "Property"
				mappings.supportedLanguage	Hard coded as constant in java file as "Language"
				mappings.supportedCodingScheme	Hard coded as constant in java file as "CodingScheme"
				mappings.supportedRepresentationalForm	Hard coded as constant in java file as "RepresentationalForm"
				mappings.supportedConceptStatus	Hard coded as constant in java file as "ConceptStatus"
				mappings.supportedPropertyLink	Hard coded as constant in java file as "PropertyLink"
				mappings.supportedPropertyQualifier	Hard coded as constant in java file as "PropertyQualifier"
				mappings.supportedDataType	Hard coded as constant in java file as "DataType"

Concepts

UMLS Mapping					
RRF File Name	***RRF Column Name***	***RRF Definition***	***NCI Meta only***	***LexGrid Model Element***	***comments***
Concepts					

MRCONSO.RRF	CODE	Unique Identifier or code for string in source		concept.conceptCode	
MRCONSO.RRF	CUI	Unique identifier for concept	x	concept.conceptCode	
				concept.isActive	Hardcoded in parameter as true.
				concept.conceptStatus	Hard coded as constant in java file as "Active"
				concept.isAnonymous	Hardcoded in parameter as false.
MRCONSO.RRF	STR	String		concept.entityDescription	
				concept.conceptProperty.Format	Hard coded as constant in java file as "text/plain" or null
				concept.conceptProperty.propertyName	May be hard coded as constant in java file as one of several properties.
				concept.conceptProperty.usageContext	
				concept.conceptProperty.propertyId	Generated value for property using "P" concatenated with a steadily incremented numerical value.
				concept.presentation.propertyId	Generated value for property textual presentation using "T" concatenated with a steadily incremented numerical value.
				concept.comment.propertyId	Generated value for property comment using "C" concatenated with a steadily incremented numerical value.
				concept.definition.propertyId	Generated value for property definition using "D" concatenated with a steadily incremented numerical value.
				concept.instruction.propertyId	Generated value for property instruction using "I" concatenated with a steadily incremented numerical value.
MRCONSO.RRF	CUI	Unique identifier for concept		concept.conceptProperty.text.content.	
				concept.conceptProperty.propertyId	Generated value for property using "CUI" concatenated with a steadily incremented numerical value.
				concept.conceptProperty.propertyName	hard coded as constant in java file as "UMLS_CUI"
				concept.conceptProperty.propertyType	hard coded as constant in java file as "property"
				concept.conceptProperty.format	left as null
MRSTY.RRF	STY	Semantic type		concept.conceptProperty.text.content	
				concept.conceptProperty.propertyId	Generated value for property using "SemType" concatenated with a steadily incremented numerical value.
				concept.conceptProperty.propertyName	hard coded as constant in java file as "Semantic_Type"
				concept.conceptProperty.propertyType	hard coded as constant in java file as "property"
				concept.conceptProperty.format	Hard coded as constant in java file as "text/plain"
MRCONSO.RRF	LAT	Language of Term(s)		concept.conceptProperty.language	Logic of code simply selects the first definition in the source as the preferred source
MRCONSO.RRF	TS	Term status		concept.presentation.isPreferred	One or a combination of these RRF values determines whether a presentation is preferred: LAT, TS, STT, ISPREF, RANK.
MRCONSO.RRF	STT	String type		concept.presentation.isPreferred	One or a combination of these RRF values determines whether a presentation is preferred: LAT, TS, STT, ISPREF, RANK.
MRCONSO.RRF	ISPREF	Indicates whether AUI is preferred		concept.presentation.isPreferred	One or a combination of these RRF values determines whether a presentation is preferred: LAT, TS, STT, ISPREF, RANK.
MRRANK.RRF	RANK	Termgroup ranking		concept.presentation.isPreferred	One or a combination of these RRF values determines whether a presentation is preferred: LAT, TS, STT, ISPREF, RANK.
				concept.presentation.isPreferred	The first presentation for each language is automatically marked as isPreferred="true" after using comparator to sort list of presentations using comparator to evaluate each presentation based on a combination of values from LAT, TS, STT, ISPREF, RANK.
MRDEF.RRF	DEF	Definition		concept.definition.text.content	
				concept.definition.isPreferred	Logic of code simply selects the first definition in the source as the preferred source
MRSAT.RRF	ATN	Attribute name		concept.conceptProperty.propertyType	Translated to a LexGrid property type. For values AN, CX, HN this property is typed as a "COMMENT" in LexGrid. For value EV this property is typed "PRESENTATION" This only occurs when the STYPE points to the CODE, SCUI or SDUI columns in MRREL.RRF or MRCONSO.RRF. If the STYPE points to SAUI then the values are loaded as property qualifiers.

MRSAT.RRF	ATV	Attribute value		concept.conceptProperty.propertyValue	
MRSAT.RRF	ATN	Attribute name		concept.conceptProperty.propertyQualifier.propertyQualifierId	If the STYPE points to SAUI then the value is loaded as a property qualifier attribute
MRSAT.RRF	ATV	Attribute value		concept.conceptProperty.propertyQualifier.content	If the STYPE points to SAUI then the value is loaded as a property qualifier attribute
MRCONSO.RRF	SAB		x	concept.conceptProperty.source.content	
			x	concept.conceptProperty.propertyQualifier.propertyQualifierId	hard coded as constant in java file as "source-code"
MRCONSO.RRF	CODE		x	concept.conceptProperty.propertyQualifier.content	
			x	concept.conceptProperty.propertyQualifier.propertyQualifierId	hard coded as constant in java file as "AUI"
MRCONSO.RRF	AUI		x	concept.conceptProperty.propertyQualifier.content	
				concept.presentation.representationalForm	When ATN value is EV this presentation will be given a representationalForm of "Abbrev."
MRCONSO.RRF	TTY	Term type in source		concept.presentation.representationForm	When TTY value is FN then representationalForm is represented as "Full Form" Otherwise the representationalForm is the same as the TTY source (i.e. if TTY is PT then representationalForm is PT.) PT is one of the preferred presentations.
				concept.conceptProperty.propertyQualifier.propertyQualifierId	hard coded as "HCD"
MRHIER.RRF	HCD	Source asserted hierarchical number or code for this atom in this context		concept.conceptProperty.propertyQualifier.content	This propertyQualifier is present when the HCD is populated in the the MRHIER file. The corresponding code and property for concept or code is qualified as a code or concept with a context derived heirarchy.

Relations

UMLS Mapping					
RRF File Name	***RRF Column Name***	***RRF Definition***	***NCI Meta only***	***LexGrid Model Element***	***comments***
Relations					
MRREL.RRF	CUI1	Unique identifier for first concept			
MRREL.RRF	AUI1	Unique identifier for first atom			
MRCONSO.RRF	CODE	Unique Identifier or code for string in source		ConceptReference.conceptCode (Model element is a ResolvedConceptReference with the value sourceOf attached to the appropriate AssociationList containing this particular REL or RELA association name.)	Mapping to the CODE depends upon the CUI or a combination of CUI and AUI values. If the CODE value is "NOCODE" then LexBIG concatenates "NOCODE" with a "-" and the CUI value. Target or source code value requires use of the DIR flag which indicates the directionality of the relationship in REL or RELA. CUI1 can be used as a pointer to the source CODE value if DIR equals Y, else CUI1 is the targetCode. Similarly, if an AUI exists AUI1 can be an indicator for CODE value to be either or source or target depending on the DIR flag.
MRREL.RRF	CUI2	Unique identifier for second concept			
MRREL.RRF	AUI2	Unique identifier for second atom			
MRCONSO.RRF	CODE	Unique Identifier or code for string in source		ConceptReference.conceptCode (Model element is a ResolvedConceptReference with the value targetOf attached to the appropriate AssociationList containing this particular REL or RELA association name.)	Mapping to the CODE depends upon the CUI or a combination of CUI and AUI values. If the CODE value is "NOCODE" then LexBIG concatenates "NOCODE" with a "-" and the CUI value. Target or source code value requires use of the DIR flag which indicates the directionality of the relationship in REL or RELA. CUI2 can be used as a pointer to the source CODE value if DIR equals Y, else CUI1 is the targetCode. Similarly, if an AUI exists AUI2 can be an indicator for CODE value to be either or source or target depending on the DIR flag.
MRREL.RRF	DIR	Source asserted directionality flag			The UMLS directional flag. Y indicates that this is the direction of the RELA relationship in its source; N indicates that it is not; otherwise indicates that it is not important or has not yet been determined. (If blank RELA, we interpret as 'N', based on empirical review of meta files).
MRREL.RRF	RELA	Relationship attribute		association.id (id inherited from Entity)	Source defined associations. If RELA value is "inverse_isa" then it is changed to "hasSubtype." All others mapped as defined in source.
MRREL.RRF	REL	Relationship		association.id (id inherited from Entity)	UMLS defined associations
		Metathesaurus			Presence of RUI in MRSAT.RRF METAUI column indicates the association defined in MRREL has an

MRSAT.RRF	METAUI	asserted unique identifier			association qualifier. Currently only MedDRA uses these.
MRSAT.RRF	ATN			AssociatedConcept.nameAndValueList.name	
MRSAT.RRF	ATV			AssociationQualification.nameAndValueList.content	
				AssociatedConcept.nameAndValueList.name	qualifier name is hard coded to "HCD" This association qualifier is attached to an association when the HCD field in MRHIER.RRF is populated. Associations are identified by evaluating a structured series of AUI's that describe the path to root (PTR field in MRHIER) Once these associations are identified they have and association qualifier attached to them with the value of the HCD loaded as the qualifier.
MRHIER.RRF	HCD			AssociationQualification.nameAndValueList.content	
MRSAB.RRF	SSN	Source short name		association.codingSchemeId (Inherited from Entity)	
MRREL.RR	REL or RELA	Relationship or Relationship attribute		association.forwardName	unqualified REL or RELA value (inverse_isa remains the same)
MRDOC.RRF	EXPL	Detailed explanation		association.reverseName	Where DOCKEY in MRDOC equals REL or RELA and value is the association name and TYPE is REL or RELA name prepended to "_inverse".
				association.inverse	Hard coded as a blank string.
				association.isAntiReflexive	hard coded to null.
				association.isAntiSymmetric	hard coded to null.
				association.isAntiTransitive	hard coded to null.
				association.isAntiTransitive	hard coded to null.
				association.isNavigable	hard coded as Boolean with value true.
				association.isReflexive	hard coded to null.
				association.isReverseFunctional	hard coded to null.
				association.isSymmetric	hard coded to null.
MRREL.RRF	SAB, REL, RELA	Source abbreviation		association.isTransitive	True when the name of the association can be mapped to a source defined in the SAB attribute of MRREL.RRF. Not the SAB value itself, but extrapolated from it using SAB to REL, RELA relationship.
				association.isTranslationAssociation	hard coded to null.
				association.targetCodingScheme	hard coded to null.
				association.entityDescription.content (inheritance path for entityDescription is Entity->versionableAndDescribable)	Hard coded to: "UMLS-defined relationships"
				relations.dc	If REL, this is hard coded as "UMLS-Relations" if RELA then it is hard coded to "Relations"
MRREL.RRF	REL, RELA		x	propertyLink.link	This is a link established when the MRREL.RRF file contains a relationship where the CUI is related to itself. Under these conditions the relationship is mapped as a property link with the MRREL defined relationship mapped as the link value.
			x	propertyLink.sourceProperty	Generated as a propertyId for concept, ex: "T-10" This is retrieved based on the AUI value in MRCONSO.RRF from the entityPropertyMultiAttrib table where the AUI equals the attributeValue column.
			x	propertyLink.targetProperty	Generated as a propertyId for concept, ex: "T-10" This is retrieved based on the AUI value in MRCONSO.RRF from the entityPropertyMultiAttrib table where the AUI equals the attributeValue column.

SNOMED UMLS Mapping

SNOMED UMLS Mapping

RRF File Name	RRF Column Name	RRF Definition	LexGrid Model Element	comments
RSAB.RRF	SVER	Release date or version number of a source	codingScheme.representsVersion	
RSAB.RRF	SSN	Source short name	codingScheme.codingScheme?	
RSAB.RRF	SON	Source Official Name	codingScheme.formalName	
		Hard coded to "en"	codingScheme.defaultLanguage	
MRSAT.RRF	ATV		concept.presentation.language	Unique to snomed.

OBO Mapping

OBO Mapping

OBO Class	OBO Entity	LexGrid Model Element	Notes
Document Header	format-version		Not mapped.
Document			Creates a codingSchemeVersion and SystemRelease record. If not

Header	data-version	CodingScheme.representsVersion	specified, then hard coded "UNASSIGNED"
Document Header	version	CodingScheme.representsVersion	Deprecated - use data-version if present.
Document Header	date		Not mapped.
Document Header	saved-by		Ignored but included if contained in the remark entity.
Document Header	auto-generated-by		Ignored but included if contained in the remark entity.
Document Header	subsetdef		Not mapped.
Document Header	import		Deprecated - Imports are used to assemble a larger document from smaller.
Document Header	typeref		Deprecated.
Document Header	synonymtypedef		Not mapped.
Document Header	idspace		Not mapped.The idspace is a triple - localName, URN and description.
Document Header	default-relationship-id-prefix		Not mapped.
Document Header	id-mapping	CodingScheme.supportedAssociation	This is more generalized than the LexGrid model, as it supports mapping between *any* id's. Note that its primary purpose, however, is to handle supportedAssociation.
Document Header	remark	CodingScheme.entityDescription	Will combine multiple remark entities into the entityDescription.
Document Header	default-namespace	codingScheme.codingScheme	Will use default-namespace if provided; otherwise will use filename without the extension.
Document Header	default-namespace	codingScheme.formalName	Will use default-namespace if provided; otherwise will use filename without the extension.
Document Header	default-namespace	codingScheme.registeredName	Combination of "urn:lsid:bioontology.org:" and if provided, the value in "default-namespace"; but if not will use filename without the extension.
		codingScheme.defaultLanguage	Hardcoded "en"
		codingScheme.isNative	Hardcoded "true"
Stanza	id	CodedEntry.conceptCode	
Stanza	name	CodedEntry.entityDescription	
		CodedEntry.presentation['textualPresentation'].text	
		CodedEntry.presentation ['textualPresentation'].isPreferred = true	
Stanza	alt_id	CodedEntry.property.property="alt_id"	
		CodedEntry.property['alt_id'].propertyId	
		CodedEntry.property['alt_id'].text	
Stanza	is_anonymous	CodedEntry.isAnonymous = true	
Stanza	is_obsolete	CodedEntry.isActive = false	
Stanza	def	CodedEntry.definition	
		CodedEntry.definition.isPreferred = true	
Stanza	def.dbxref		See dbxref
Stanza	comment	CodedEntry.comment.text	
Stanza	subset	property[subset tag]	See subsetdef
Stanza	syonym	presentation['textualPresentation'].text	
Stanza	synonym.scope	presentation ['textualPresentation'].degreeOfFidelity	
Stanza	synonym.type	presentation ['textualPresentation'].representationalForm	
Stanza	synonym.dbxref	(see dbxref)	
Stanza	exact_synonym		See synonym
Stanza	narrow_synonym		See synonym
Stanza	broad_synonym		See synonym
Stanza	xref	associations.['mapsTo']	
Stanza	xref_analog		See synonym
Stanza	xref_unk		
Stanza	is_a	associations.['hasSubtype']	Reverse of the source and target.
Stanza	is_a.namespace		If present, the supplied namespace becomes the owning "codingScheme".
Stanza	is_a.derived	associations.hasSubtype.associationQualifier	If present, need to include derived in the supportedAssociationQualifiers section
Stanza	intersection_of		Processed the same way that OWL intersection operator is processed. This includes creation of anonymous sets.
Stanza	union_of		Same as OWL
Stanza	disjoint_from		Same as OWL
Stanza	relationship	associations.	

Stanza	relationship.not_necessary	associations..associationQualifier	
Stanza	relationship.inverse_necessary	associations..associationQualifier	
Stanza	relationship.namespace		If present, the supplied namespace becomes the owning "codingScheme".
Stanza	relationship.derived	associations..associationQualifier	
Stanza	relationship.cardinality	associations..associationQualifier	
Stanza	relationship.maxCardinality	associations..associationQualifier	
Stanza	relationship.minCardinality	associations..associationQualifier	
Stanza	is_obsolete	codedEntry.isActive = false	
		codedEntry.conceptStatus="is_obsolete"	
Stanza	replaced_by		
Stanza	consider		Not Mapped
Stanza	use_term		(deprecated)
dbxref	dbxref name	CodedEntry..source	
		supportedSource	dbxref name format is inconsistent. In most cases, it can be the localName of supportedSource, but special processing may be necessary in the case of URL's, etc
dbxref	dbxref description		Not mapped.
dbxref	trailing modifiers		Not mapped.
typeDef Stanza	domain	associations.['has_domain']	
typeDef Stanza	range	associations.['has_range']	
typeDef Stanza	is_cyclic	property['is_cyclic']	
typeDef Stanza	is_reflexive	property['is_reflexive']	
		association.isReflexive	
typeDef Stanza	is_symmetric	property['is_symmetric']	
		association.isSymmetric	
typeDef Stanza	is_transitive	property['is_transitive']	
		association.isTransitive	
typeDef Stanza	inverse_of	association.inverse	
instance stanza	id	same rules as general stanza	same rules as general stanza
instance stanza	name	same rules as general stanza	same rules as general stanza
instance stanza	instance_of	association['has_instance']	
instance stanza		CodedEntry.property.property=""	data type properties go in Coded Entry property section

HL7 RIM Mapping

		HL7 RIM Mapping	
HL7 Table	**HL7 Column**	**LexGrid Model Element**	**Notes**
Model	<modelID>	<codingSchemeName>	
	<name>	<formalName>	
		<registeredName>	http://www.hl7.org/Library/data-model/RIM *[1]
		<defaultLanguage>	en*
	<versionNumber>	<representsVersion>	
		<isNative>	0*
		<approximateNumberofConcepts>	Result of count on concept beari table?
		<firstRelease>	MISSING
		<modifiedRelease>	MISSING
		<deprecated>	MISSING
	<description>	<entityDescription>	
		<copyright>	MISSING
VCS_code_system	codeSystemId	codingScheme.registeredName	Moved to metadata file.
	codeSystemType	commonTypes::Properties	This is an HL7 specific code sys property to distinguish internal v external code systems. Moved tc metadata file.

	codeSystemName	concept.conceptCode	Moved to metadata file.
	codeSystemName	concept.presentation ['textualPresentation'].text	
	fullName	codingScheme.formalName	
	description	codingScheme.entityDescription	Moved to metadata file.
	releaseId	codingScheme.representsVersion	Moved to metadata file.
	copyrightNotice	codingScheme.copyright	Moved to metadata file.
	literal('en')	codingScheme.defaultLanguage	Moved to metadata file.
VCS_concept_code_xref	internalId		
	Concept Code	concept.conceptCode	
	Case Difference	commonTypes::Properties	Basically a proprty to outline wh there are case differences in the Concept Code or not (mainly use but not restricted tor units of measure)
	Status	concept.isActive=(conceptStatus=='A'?)	
		concept.conceptStatus	Not used by HL7. A = isActive, retired
VCS_concept_designation	internalId		foreign key
	designation	concept.presentation ['textualPresentation'].text	
	designationSeq		
	language	concept.presentation ['textualPresentation'].language	Can be omitted if language = def language
	preferredForLanguage	concept.presentation ['textualPresentation'].isPreferred	
VCS_concept_description	internalId	with(codeSystem[deref(internalId)].concept [deref(internalId)]).definition	foreign key
	description	concept.presentation ['textualPresentation'].text	
	language	concept.presentation ['textualPresentation'].language	
	literal('true')	concept.presentation ['textualPresentation'].isPreferred	
	uniqueId()	concept.presentation ['textualPresentation'].propertyId	
	literal('definition')	concept.presentation ['textualPresentation'].property	
VCS_concept_property	internalId		foreign key
	propertyCode	concept.property.property	
	propertySeq		Currently not used by HL7
	propartyValue	concept.property.text	
	language	concept.property.language	
VCS_concept_relationship	relationCode	association.association	
	sourceInternalId	associationInstance.sourceConcept	
	targetInternalId	associationTarget.targetConcept	
Model	modelID	systemRelease.releaseId	
	name	service.service	
	versionNumber	service.version	
	lastModifiedDate	systemRelease.releaseDate	
	developingOrganization	systemRelease.releaseAgency	
	committeeID		
	description	systemRelease.entityDescription	
	concat ('urn:oid:2.16.840.1.113883:',systemRelease.releaseId)	systemRelease.releaseURN	
	literal('true')	systemRelease.isLatest	Also have to set the prior release isLatest to false
	preceding-sibling/releaseOrder + 1	systemRelease.releaseOrder	
Model	modelID	commonTypes::Properties	
(Special mapping for NCI)	name	codingScheme.localName	
	versionNumber	codingScheme.representsVersion	
	lastModifiedDate	commonTypes::Properties	
	developingOrganization	commonTypes::Properties	
	committeeID		
	description	codingScheme.entityDescription	
	concat	codingScheme.registeredName	

	('urn:oid:2.16.840.1.113883:',systemRelease.releaseId)		
	literal('true')	commonTypes::Properties	Also have to set the prior release isLatest to false
	preceding-sibling/releaseOrder + 1	commonTypes::Properties	
RIM_vocabulary_domain	vocDomain	codingscheme ["VocabularyDomain"].concept.conceptCode	Vocabulary Domains are carried code system of vocabulary doma
		codingscheme ["VocabularyDomain"].concept.presentation ["textualPresentation"].text	preferredPresentation
	description	codingscheme ["VocabularyDomain"].concept.definition.text	preferredDefinition for code
	restrictsDomain	codingscheme ["VocabularyDomain"].association ["hasSubtype"].sourceConcept	*Should this be hasSubtype or something else?*
		codingscheme ["VocabularyDomain"].association ["hasSubtype"].targetconcept = vocDomain	
VOC_code_reference	usedToBuildValueSet	with(valueDomain[registeredName=current ()/.])	
	referencesConceptCode	…valueDomainEntry/conceptCode	1) id is synthesized 2) Only stored if isHeadCode == false or includeReferencedCode true
	referencesInternalId		Internal id's aren't exposed in lex
	relationship	…valueDomainEntry/includeChildren = (relationship == 'hasSubtype')	Won't deal w/ non-hasSubtype relationships, but HL7 doesn't ha any.
	includeReferencedCode	…valueDomainEntry/isSelectable	
	leafOnly		Not used in HL7 Model
	directChildrenOnly		Not used in HL7 Model
	isHeadCode		Only used when referenced in VOC_value_set_constructor.
	referencesCodeSystem	…/valueDomainEntry.codingScheme	Shortcut in HL7 model. Must = VOC_value_set.basedOnCodeSy
	arbitraryUniqueValue()	…/valueDomainEntry.id	
VOC_registered_code_system	codeSystemId		VOC_registered_code_system is currently transferred to Lexgrid
	sponsor		
	publisher		
	versionReportingMethod		
	licensingInformation		*This field should really be transf copyright?*
	inUMLS		
	systemSpecificLocatorInfo		
	uri		
	isExternal		
VOC_value_set	valueSetId	valueDomain.registeredName	
	valueSetName	valueDomain.valueDomain	*Name is the key in LexGrid, and optional in HL7 - will need to be addressed.*
	basedOnCodeSystem	valueDomain.defaultCodingScheme	*Optional in HL7, required in LexGrid.*
	description	valueDomain.entityDescription	
	definingExpression		Not used.
	allCodes	if 'true': valueDomain.conceptCode = "@", valueDomain.includeChildren='true'	
	isTaxonomicSet		No mapping available
	valueSetAuthority		Included in valueSetID
	valueSetNumber		
VOC_value_set_constructor	usedToBuildValueSet	new valueDomainEntry(parent = valueDomain[valueSetId=current ()/.],id=unique())	
	includesOrExcludesSet	valueDomainEntry.includesValueDomain	
	includeHeadCode	valueDomainEntry.isSelectable	
		valueDomainEntry.conceptCode = VOC_code_reference [usedToBuildValueSet=current ().usedToBuildValueSet and isHeadCode=true].referencesConceptCode	*Assumes that there always is a h code.*

VOC_vocabulary_domain_value_set	representsVocDomain	(selector)	
	definedByValueSet	codingscheme['VocabularyDomain'].concept [representsVocDomain].property ['definedByValueSet'].text	*have to get 'representsVocDoma into supportedProperty header*
	appliesInContext	codingscheme['VocabularyDomain'].concept [representsVocDomain].property ['definedByValueSet'].usageContext	*Have to get all the contexts in th VocabularyDomain supportedContext header*
VCS_release_version	releaseId	codingSchemeVersion.version	Note: this is not the way that thi are done at the moment. At the moment, VCS_release_versions loaded into systemRelease. Ente one or more concept/relationshi change.
		valueDomainVersion.version	Set iff one or more value sets ch
	literal("false")	codingSchemeVersion.isComplete	All versions are delta's in this m
	releaseAgency		
	releaseDate	codingSchemeVersion.versionDate	
		valueDomainVersion.versionDate	
	description	codingSchemeVersion.entityDescription	
		valueDomainVersion.entityDescription	
	editorID		There is no place for these curre
	forWhomID		
	concat ('urn:oid:2.16.840.1.113883:',systemRelease.releaseId)		This corresponds to the containi system release when the sytem release occurs. It is empty until t

LexGrid Text Mapping

LexGrid Text Mapping								
		Source Definition						
		Column	1	2	3	4	5	6
Line	1		<codingSchemeName>	<codingSchemeId>	<defaultLanguage>	<formalName>	[<version>]	[

```
] || [<description>] || [<copyright>] ||This must be the first l
|-
!style="background:#FF9900;"|
|style="background:#FF9900;"|2
||| <nowiki>[<code>]</nowiki> ||<name>||[<description>]|||||||||
|-
!style="background:#FF9900;"|
|style="background:#FF9900;"|3
||||| <nowiki>[<code>]</nowiki> ||<name>||[<description>]|||||||
|-
| |||||||||||||||||||||
|-
| |||||||||||||||||||||
|-
| |||||||||||||||||||||
|-
| |||||
! style="background:#3399FF;"|Text Element
! style="background:#3399FF;"|LexGrid
! style="background:#3399FF;"|Comments
||
||
||
||
||
||
|-
| |||||
! style="background:#FFFF99;"|Coding Scheme
! style="background:#FFFF99;"|
! style="background:#FFFF99;"|
||
||
||
||
||
||
|-
| ||||||codingSchemeName||codingScheme.codingSchemeName|||||||||
|-
| ||||||codingSchemeId||codingScheme.codingSchemeId||||||||||||
|-
| ||||||defaultLanguage||codingScheme.defaultLanguage||||||||||
|-
| ||||||formalName||codingScheme.formalName|||||||||||||
|-
| ||||||version||codingScheme.representsVersion||Optional|||||||
|-
| ||||||source||codingScheme.source||Optional|||||||||||
|-
| ||||||description||codingScheme.entityDescription||Optional|||
|-
| ||||||copyright||codingScheme.copyright||Optional|||||||||||
|-
| |||||
! style="background:#FFFF99;"|Concepts
! style="background:#FFFF99;"|
! style="background:#FFFF99;"|
||
||
||
||
||
||
|-
| ||||||code||concept.conceptCode||Optional|||||||||||
|-
| ||||||name||concept.conceptName|||||||||||||
|-
| ||||||description||concept.entityDescription|||||||||||||
|-
|
|}
```

Retrieved from "https://cabig-kc.nci.nih.gov/Vocab/KC/index.php/LexEVS_5.x_Loader_Model_Elements_Mapping"
Categories: VKC Contents | Documentation | LexEVS

- This page was last modified on 31 January 2010, at 04:27.

National Cancer Institute
U.S. National Institutes of Health | www.cancer.gov
caBIG Knowledge Center
A part of the Enterprise Support Network

- Home
- Knowledge Centers
  - caGrid
  - Clinical Trials Management Systems
  - Data Sharing and Intellectual Capital
  - Molecular Analysis Tools
  - Tissue/Biospecimen Banking and Technology Tool
  - Vocabulary
- Discussion Forums
  - caBIG General Forum
  - caGrid
  - Clinical Trials Management Systems
  - Data Sharing and Intellectual Capital
  - Molecular Analysis Tools
  - Tissue/Biospecimen Banking and Technology Tool
  - Vocabulary
- Bugs/Feature Requests
- Development Code Repository

LexEVS 5.x Loader Source Mapping

From Vocab_Wiki

LexEVS 5.x Loader Model Elements Mapping > Main Page > LexEVS 5.x Loader Guide > Main Page > LexEVS 5.x Loader Source Mapping

Contents

Introduction

This document is a section of the Loader Guide. It was formerly the LexEVS v5.0 *Source Mapping Guide*.

NCI MetaThesaurus: RRF content (v5.1)

In LexEVS v5.1, loader enhancements for RRF content were made. Loads of the NCI MetaThesaurus RRF formatted data into the LexGrid model now accurately reflect the state of the data as it exists in the current RRF files. In the text that follows, the **Problem** sections describe LexEVS v5.0 behavior, and the **Solution** sections describe LexEVS v5.1 behavior.

Data Model Elements

Most data elements will be loaded as either properties or property qualifiers.

[https://wiki.nci.nih.gov/download/attachments/18947057/Property.jpg]

A few will be loaded as qualifiers to associations.

MRREL.RRF File

Problem:

REL and RELA column elements from the RRF source need to be connected. Currently these are loaded as separate relationships preventing the user from connecting to the REL/RELA combinations that actually occur in the NCI-META (e.g. RELA may be different for same REL value in different sources).

Requirement:

A single relationship should be loaded for a REL/RELA combination for a particular SAB between two CUIs.

Solution:

Since RELA type RRF elements have been defined as relationship names specific to sources and not independent relationships themselves, these elements will be loaded as association qualifiers in the LexGrid model.

Problem and Requirement:

User is unable to distinguish individual relationships from one source or another. The same association "entity" exists only once but has two "source" qualifiers. User is unable to distinguish the AUI1/STYPE1 and AUI2/STYPE2 which gives us the information about what source data structures are actually being connected by MRREL entries. Users also need the ability to associate AUI/STYPE fields with SAB. Users sole choice for rendering a relationship in terms of the strings on either side is to use preferred concept names.

Proposed Solution:

Propose AUI to AUI - the way CUI to CUI are currently handled in the implementation. Propose entity to entity relationship - will still have to account for CUI to CUI relationships. Load each unique RUI (would be quite large). They would need to be listed as supported association (this is not traditional how it is used).

Load supporting column elements from MRREL.RRF including contents of: AUI1, STYPE1, AUI2, STYPE2, SRUI, SAB, RG, SUPPRESS, CVF, RUI

These will be available as elements of the overriding Metathesaurus Association and loaded as association qualifiers.

Problem:

Self Referencing Relationships (CUI1 = CUI2) cannot be fully represented in our model. Previously, these were loaded as PropertyLinks. This fit into the LexEVS model well, but left out important RRF information. Most notably, PropertyLinks cannot contain Qualifiers like normal relations can. Because of the increased number of Qualifiers that are required to be placed on relations, much information would be lost representing these relations as PropertyLinks

Solution:

Do not treat a CUI1 = CUI2 relationships differently than a CUI1 != CUI2 relationship. For API and query purposes, qualify these relationships with a 'selfReferencing=true' Qualifier. In this way, we can still avoid cycles in the API, but maintain all relevant Qualifier information in the relation.

MRSAT.RRF

Problem:

MRSAT.RRF is not loaded but only accessed for given preferred term algorithms. This data should be loaded as concept properties (STYPE=CUI), properties on properties (STYPE=AUI, SAUI, CODE, SCUI, SDUI), qualifiers on associations (STYPE=RUI,SRUI). Some complexity may arise as concept properties can have additional qualifiers, but property-properties cannot and association-qualifiers cannot.

Requirement:

If the STYPE is something other than RUI or SRUI, you can load that row as an entity property. The fields you'd want to capture are:

CUI - We use this as the entityCode and is loaded as such in the table.

METAUI - load as a propertyQualifier (name=METAUI, value)

STYPE - load as a propertyQualifier (name=STYPE, value)

ATUI - load as propertyId

ATN - load as property name

SAB - load as a propertyQualifier (typeName=source)

ATV- load as a propertyValue

SUPPRESS - load as propertyQualifier if value != N

MRRANK.RRF

Problem:

SAB specific ranking of representational form in MRRANK is not exposed to the user (used in an underlying ranking and specifying of preferred presentations for a given concept)

Requirement:

Load elements of MRRANK so that they are available to the user.

Proposed Solution:

Load MRRANK as property qualifier on Presentation type property with the property Name of "mrrank."

Retrieval:

Available in current LexEVS api

MRSAB.RRF

Problem:

MRSAB.RRF file data is not loaded or is otherwise unavailable to the user.

Requirement:

Load MRSAB.RRF file data as metadata

Implemented Solution:

Entire content of each row of MRSAB file is loaded as metadata to an external xml file with tags created from column names and value inserted between tags as is appropriate

MRMAP.RRF, MRSMAP.RRF

Problem:

MRMAP.RRF source load is not supported in current load. Currently this RRF file is not populated in NCI Metathesaurus distributions. Mapping is not explicitly supported in the LexGrid Model.

Requirement:

Load MRMAP data.

Solution:

To be evaluated for a load to current model elements or possible new model mapping elements. The general agreement is that this is more appropriately implemented in 6.0.

MRHIER.RRF

Problem:

HCD is loaded as a property on the presentation but the SAB isn't associated with it so we do not know the source of the HCD. (only look at row that has HCD field populated) Path to Root, (PTR) is also not loaded, but is instead used to determine path to root operations in LexEVS.

Requirement:

These elements need to be loaded and available from the LexEVS api

Solution:

Load HCD associated field SAB as property qualifier when HCD is present. Load PTR as property.

MRDOC.RRF

Problem:

MRDOC contains metadata unavailable to the user. It is not loaded by LexEVS.

Requirement:

This metadata will be made available to the user.

Solution:

MRDOC's column names and content will be processed as tag/value mappings to a metadata file.

MRDEF.RRF

Problem:

Some values from each row are not loaded by LexEVS.

Requirement:

AUI should be loaded to connect it with the presentation ATUI, SUPPRESS, CVF, SATAUI should be loaded and exposed to the user.

ATUI, SUPPRESS, CVF, SATAUI, column values will be loaded as property qualifiers on the Definition type property derived from MRDEF column.

MRCONSO.RRF

Problem:

Some elements from the columns of MRCONSO.RRF are not loaded by LexEVS.

Requirement:

Load LUI, SUI, SAUI, SDUI, SUPPRESS, CVS fields and expose to the user.

Solution:

All noted values will be loaded as property qualifiers.

Unified Medical Language System

The Unified Medical Language System (UMLS) and Rich Release Format (RRF) files

The UMLS' large medical thesaurus is available as a set of text based, "|' separated files which can be made subset into individual terminologies depending on the user's needs. NCI's MetaThesaurus is also RRF formatted. We map individual terminologies, the entire NCI MetaThesaurus and the UMLS terminology SEMNET into LexGrid Using specific loaders and mappings for each.

Supported Coding Scheme Attributes:

> These aren't mapped as categories to a model element. That is, a supported association has an attributeTag column with a corresponding name, but it's context is implied in the name of the supported attribute. For instance, supported associations will have an attributeTag of "association" but that tag corresponds to no element in the model element SupportedAssociation. Instead the context is implied in the name of the element SupportedAssociation.

Preferred Presentation Selection:

Preferred Presentation is determined first by sorting the presentations to include first those in the default language of the Terminology. Following that and given there is more than one presentation in the default language the “most preferred” is determined in the following manner:

Using the "isPref" column, the "TS" and "STT" columns in the MRCONSO RRF file, or a combination of these columns. The MRRANK file overrides these columns.

Preferred Definition Selection:

Definitions in UMLs are not ranked, the first definition found for a concept in the source file MRDEF.RRF is set to preferred.

Special SNOMED adjustments for concept presentation language:

Snomed handles it’s language default settings differently than other UMLS terminologies, we hard code it’s default language as “en” as a result.

Presentation language is determined by combining the values of SUI, LUI and CUI from MRCONSO and selecting the ATV value from MRSAT where SAB always equals SNOMEDCT and the ATN value is either LANGUAGECODE or SUBSETLANGUAGECODE.

Association Qualifiers for medDRA and others:

MedDRA employs SMQ’s or Standardized Medical Queries as a method of classifying portions of this terminology. These are expressed in MRSAT.RRF when the AUI in the METAUI column is replaced by a RUI code. In LexBIG is RUI is identified in the MRREL.RRF source as relationships are loaded and the associated ATN and ATV values from the MRSAT.RRF row are populated as association qualifier name and value.

Hierarchies expressed in source contexts:

Hierarchies in the UMLS are expressed in the MRREL.RRF file as source, target pairs. However source hierarchies may also be expressed in the MRHEIR.RRF file. These context based hierarchies are realized in LexBIG by accessing the MRHEIR source where the HCD column value is populate. When this is the case, as in MESH, the path of AUI’s to root from the code in the HCD column is processed as a hierarchy. LexBIG’s behavior is as follows:

- Entries in MRHIER that define multiple contexts (HCD field) per CUI will trigger additional tracking within the LexBIG environment.
- Each link is tracked via the corresponding contextual chain(Path To Root field). To do this, we add association qualifiers that tag the association between each participating concept. The qualifier name is 'HCD' and the value will be the HCD field value from the MRHIER file.
- An individual association between two concepts can participate in multiple context chains by assigning additional association qualifiers. A complete flow across the entire chain of links (essentially reconstructing PTR field) can be derived by recursive evaluation of surrounding links that have the same context qualifications. Since each

concept can carry multiple text presentations, property qualifiers will be used to track the individual terms used in each context.
- As with associations, multiple qualifiers can be assigned to each text property. Once again, the qualifier name will be 'HCD' and the value will be the HCD field value from the MRHIER file.
- In order to query context-specific relationships, we can first use the API to filter the relationships a concept participates in, then query neighboring nodes to determine the complete context path, and finally map back to specific terms through the registered HCD qualifiers .

OBO Mapping

The OBO each remark in the document header will be combined and put into the coding scheme entityDescription.

For example:

```
remark: autogenerated-by:     DAG-Edit version 1.320
remark: saved-by:             mariacos
remark: date:                 Fri Jun 27 09:41:28 EDT 2003
remark: version: $Revision: 1.1 $
```

Protege OWL

DatatypeProperty Representation

Owl:

```
<owl:DatatypeProperty rdf:ID="currency">
          <rdfs:domain rdf:resource="#Money"/>
          <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
     </owl:DatatypeProperty>
```

In LexGrid, a DatatypeProperty is combination of a conceptProperty and Assocation.

Concept Property

```
<lgCon:concept id="Money">
      <lgCommon:entityDescription>Money</lgCommon:entityDescription>
      ….
     <lgCon:conceptProperty propertyId="P0003" propertyName="currency">
        <lgCommon:text>xsd:string</lgCommon:text>
      </lgCon:conceptProperty>
    </lgCon:concept>
```

Association

```
<lgRel:association id="hasDomain" forwardName="hasDomain" isReflexive="false" isSymmetric="false"
isTransitive="true" reverseName="kindIsDomainOf">
      <lgRel:sourceConcept sourceEntityType="association" sourceId="currency">
        <lgRel:targetConcept targetEntityType="concept" targetId="Money"/>
      </lgRel:sourceConcept>

 <lgRel:association id="currency">
      <associationProperty propertyId="P0007" propertyName="isDatatypeProperty">
        <lgCommon:text>true</lgCommon:text>
      </associationProperty>
      <associationProperty propertyId="P0008" propertyName="isObjectProperty">
        <lgCommon:text>false</lgCommon:text>
      </associationProperty>
    </lgRel:association>

 <lgRel:association id="datatype" forwardName="datatype">
      <lgRel:sourceConcept sourceEntityType="association" sourceId="currency">
        <lgRel:targetDataValue dataId="D0001">
          <lgRel:dataValue>string</lgRel:dataValue>
        </lgRel:targetDataValue>
```

Equivalent Class Representation

Owl:

```
<owl:Class rdf:ID="Father">
    <owl:equivalentClass>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#Person"/>
          <owl:Restriction>
            <owl:onProperty>
              <owl:FunctionalProperty rdf:about="#hasSex"/>
            </owl:onProperty>
            <owl:hasValue rdf:resource="#MaleSex"/>
          </owl:Restriction>
          <owl:Restriction>
            <owl:someValuesFrom rdf:resource="#Person"/>
            <owl:onProperty>
              <owl:ObjectProperty rdf:about="#hasChild"/>
            </owl:onProperty>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </owl:equivalentClass>
  </owl:Class>
```

In LexGrid, the equivalentClass is represented as an Association.

Association

```
<lgRel:association id="equivalentClass" forwardName="equivalentClass" isReflexive="true" isSymmetric="
  <lgRel:sourceConcept sourceEntityType="concept" sourceId="Father">
        <lgRel:targetConcept targetEntityType="concept"     targetId="A38"/>
```

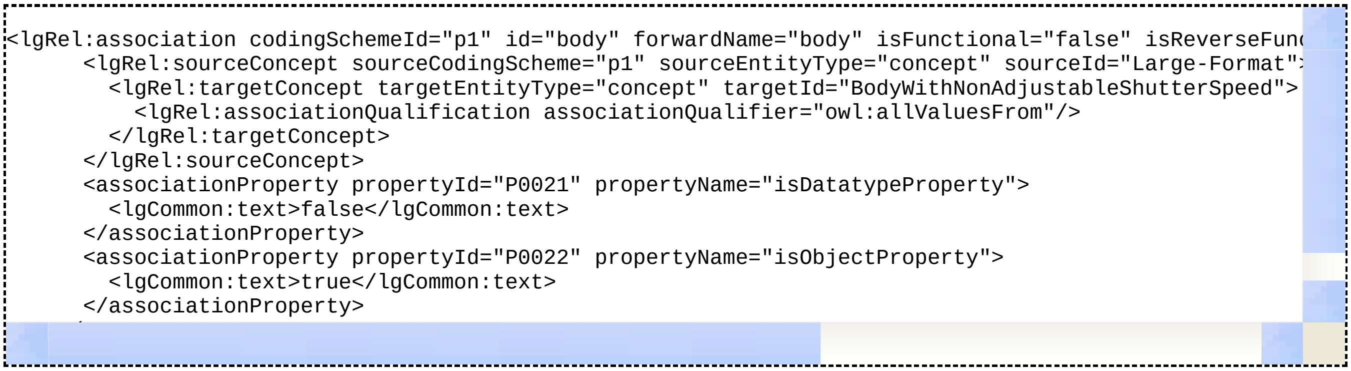
Restriction Representation

Owl:

```
<owl:Class rdf:ID="Large-Format">
          <rdfs:subClassOf rdf:resource="#Camera"/>
          <rdfs:subClassOf>
               <owl:Restriction>
                     <owl:onProperty rdf:resource="#body"/>
                     <owl:allValuesFrom rdf:resource="#BodyWithNonAdjustableShutterSpeed"/>
               </owl:Restriction>
          </rdfs:subClassOf>
     </owl:Class>
```

In LexGrid, a restriction is a combination of association and qualifier.

Association:



```
<lgRel:association codingSchemeId="p1" id="body" forwardName="body" isFunctional="false" isReverseFunc
      <lgRel:sourceConcept sourceCodingScheme="p1" sourceEntityType="concept" sourceId="Large-Format">
        <lgRel:targetConcept targetEntityType="concept" targetId="BodyWithNonAdjustableShutterSpeed">
          <lgRel:associationQualification associationQualifier="owl:allValuesFrom"/>
        </lgRel:targetConcept>
      </lgRel:sourceConcept>
      <associationProperty propertyId="P0021" propertyName="isDatatypeProperty">
        <lgCommon:text>false</lgCommon:text>
      </associationProperty>
      <associationProperty propertyId="P0022" propertyName="isObjectProperty">
        <lgCommon:text>true</lgCommon:text>
      </associationProperty>
```

Additional Examples

Owl:

```
<owl:Class rdf:ID="Father">
    <owl:equivalentClass>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#Person"/>
          <owl:Restriction>
            <owl:onProperty>
              <owl:FunctionalProperty rdf:about="#hasSex"/>
            </owl:onProperty>
            <owl:hasValue rdf:resource="#MaleSex"/>
          </owl:Restriction>
          <owl:Restriction>
            <owl:someValuesFrom rdf:resource="#Person"/>
            <owl:onProperty>
              <owl:ObjectProperty rdf:about="#hasChild"/>
            </owl:onProperty>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </owl:equivalentClass>
  </owl:Class>
```

LexGrid:

```
<lgRel:association id="equivalentClass" forwardName="equivalentClass" isReflexive="true" isSymmetric="
  <lgRel:sourceConcept sourceEntityType="concept" sourceId="Father">
        <lgRel:targetConcept targetEntityType="concept"     targetId="A38"/>
      </lgRel:sourceConcept>


 <lgRel:association codingSchemeId="" id="hasSex" forwardName="hasSex" isFunctional="true" isReverseFu
    <lgRel:sourceConcept sourceEntityType="concept" sourceId="A38">
        <lgRel:targetConcept targetEntityType="concept" targetId="MaleSex">
          <lgRel:associationQualification associationQualifier="owl:hasValue"/>
        </lgRel:targetConcept>

<lgRel:association codingSchemeId="rdfs" id="subClassOf" forwardName="subClassOf" isFunctional="false"
   <lgRel:sourceConcept sourceEntityType="concept" sourceId="A38">
        <lgRel:targetConcept targetEntityType="concept" targetId="Person"/>
      </lgRel:sourceConcept>

 <lgRel:association codingSchemeId="" id="hasChild" forwardName="hasChild" isFunctional="false" isReve
   <lgRel:sourceConcept sourceEntityType="concept" sourceId="A38">
        <lgRel:targetConcept targetEntityType="concept" targetId="Person">
          <lgRel:associationQualification associationQualifier="owl:someValuesFrom"/>
        </lgRel:targetConcept>

<lgCon:concept id="A38" isAnonymous="true">
      <lgCommon:entityDescription>Person and (hasSex has MaleSex) and (hasChild some Person)</lgCommon
      <lgCon:presentation propertyId="P0002" propertyName="textualPresentation" isPreferred="true">
        <lgCommon:text>Person and (hasSex has MaleSex) and (hasChild some Person)</lgCommon:text>
      </lgCon:presentation>
      <lgCon:conceptProperty propertyId="P0001" propertyName="type">
        <lgCommon:text>owl:intersectionOf</lgCommon:text>
      </lgCon:conceptProperty>
```

Property Restriction Representation

Anonymous LexGrid concepts are created for property restrictions (UnionOf, hasValue).

Example 1

Owl:

```
<owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#Hot"/>
          <owl:Class rdf:ID="Medium"/>
          <owl:Class rdf:about="#Mild"/>
        </owl:unionOf>
      </owl:Class>
```

LexGrid:

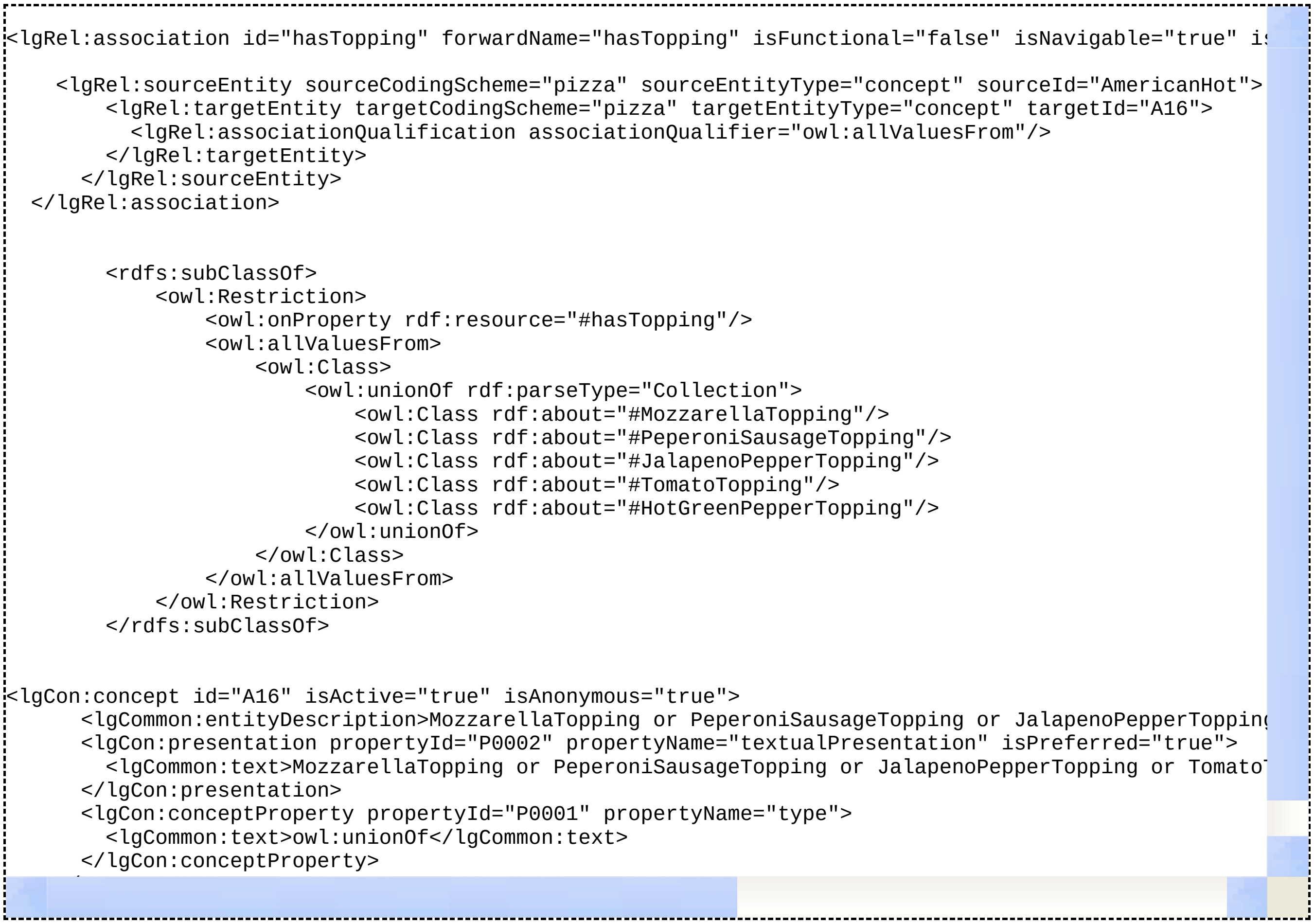
```
<lgCon:concept id="A17" isAnonymous="true">
      <lgCommon:entityDescription>Hot or Medium or Mild</lgCommon:entityDescription>
      <lgCon:presentation propertyId="P0001" propertyName="textualPresentation" isPreferred="true">
        <lgCommon:text>Hot or Medium or Mild</lgCommon:text>
      </lgCon:presentation>
      <lgCon:conceptProperty propertyId="P0002" propertyName="isUnion">
        <lgCommon:text>true</lgCommon:text>
      </lgCon:conceptProperty>
      <lgCon:conceptProperty propertyId="P0003" propertyName="isIntersection">
        <lgCommon:text>false</lgCommon:text>
      </lgCon:conceptProperty>
      <lgCon:conceptProperty propertyId="P0004" propertyName="isEnumeration">
        <lgCommon:text>false</lgCommon:text>
      </lgCon:conceptProperty>
    </lgCon:concept>
```

Example 2

Owl:

```
           <owl:Restriction>
                <owl:onProperty rdf:resource="#hasTopping"/>
                <owl:allValuesFrom>
                    <owl:Class>
                        <owl:unionOf rdf:parseType="Collection">
                            <owl:Class rdf:about="#MozzarellaTopping"/>
                            <owl:Class rdf:about="#PeperoniSausageTopping"/>
                            <owl:Class rdf:about="#JalapenoPepperTopping"/>
                            <owl:Class rdf:about="#TomatoTopping"/>
                            <owl:Class rdf:about="#HotGreenPepperTopping"/>
                        </owl:unionOf>
                    </owl:Class>
                </owl:allValuesFrom>
            </owl:Restriction>
```

LexGrid:



```
<lgRel:association id="hasTopping" forwardName="hasTopping" isFunctional="false" isNavigable="true" i

    <lgRel:sourceEntity sourceCodingScheme="pizza" sourceEntityType="concept" sourceId="AmericanHot">
        <lgRel:targetEntity targetCodingScheme="pizza" targetEntityType="concept" targetId="A16">
           <lgRel:associationQualification associationQualifier="owl:allValuesFrom"/>
        </lgRel:targetEntity>
      </lgRel:sourceEntity>
  </lgRel:association>


        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#hasTopping"/>
                <owl:allValuesFrom>
                    <owl:Class>
                        <owl:unionOf rdf:parseType="Collection">
                            <owl:Class rdf:about="#MozzarellaTopping"/>
                            <owl:Class rdf:about="#PeperoniSausageTopping"/>
                            <owl:Class rdf:about="#JalapenoPepperTopping"/>
                            <owl:Class rdf:about="#TomatoTopping"/>
                            <owl:Class rdf:about="#HotGreenPepperTopping"/>
                        </owl:unionOf>
                    </owl:Class>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>


<lgCon:concept id="A16" isActive="true" isAnonymous="true">
      <lgCommon:entityDescription>MozzarellaTopping or PeperoniSausageTopping or JalapenoPepperToppin
      <lgCon:presentation propertyId="P0002" propertyName="textualPresentation" isPreferred="true">
        <lgCommon:text>MozzarellaTopping or PeperoniSausageTopping or JalapenoPepperTopping or Tomato
      </lgCon:presentation>
      <lgCon:conceptProperty propertyId="P0001" propertyName="type">
        <lgCommon:text>owl:unionOf</lgCommon:text>
      </lgCon:conceptProperty>
```

NCI OWL

Top-level containers for relations are created, which separate the association types based on the notion of ‘associations’ and ‘roles’ as defined by NCI:

- Associations are “non-inheritable, non-defining relations between concepts”
- Roles are “inheritable relationships”

A LexGrid concept is created for every anonymous class present in the OWL ontology.

If no equivalent class for a concept, it is considered primitive and is indicated by creating a concept property set to ‘true.’

Embedded XML

Property text with embedded XML fragments are identified by by the following identifiers:

qual-name qual-value qual

If the extracted tag is one of XML Text identifiers:

Value term-name def-definition go-term

The text of the property is set to the tag value.

If the extracted tag is one of XML Source Name identifiers:

term-source def-source

A property source is created and the tag value identifies the source.

If the property is a presentation and the extracted tag is XML Representational Form:

term-group

The representational form of the presentation property is set to the tag value.

If the extracted tag is one of DB XRef Prefix:

dbxref.*

A property qualifier is created. The property qualifier id is set to the tag, the value is set to the tag value.

HL7 RIM

To build a single coding scheme from the HL7 MS Access database, implementation is similar to how the NCI MetaThesaurus is stored in LexGrid.

For example, here is how entries MTHU021347 and MTHU033458 in ICPC2ICD10ENG (NCI MethThesaurus C1394796) are structured in LexGrid:

Coding Scheme: NCI MetaThesaurus - urn:oid:2.16.840.1.113883.3.26.1.2

Concept Code: C1394796

Entity Description: decompensation; heart, senile

Status: Active

Is Active: true

Is Anonymous: false

Presentation: decompensation; heart, senile

> **Property Name:** textualPresentation
>
> **Property Id:** T-1

Language: ENG

Is Preferred: true

Representational Form: PT

Source: ICPC2ICD10ENG , **Role:** null, **SubRef:** null

Property Qualifier Id: source-code , **Property Qualifier Content:** MTHU021347

Presentation: heart; decompensation, senile

Property Name: textualPresentation

Property Id: T-2

Language: ENG

Is Preferred: false

Representational Form: PT

Source: ICPC2ICD10ENG , Role: **null, SubRef: null**

Property Qualifier Id: source-code , **Property Qualifier Content:** MTHU033458

ConceptProperty: Mental or Behavioral Dysfunction

Property Name: Semantic_Type

Property Id: SemType-1

In HL7, code systems, concepts, and designations are in the following tables:

Table: VCS_concept_code_xref

Internal concept identifier	Code system	OID	Concept code	Case difference Status
10011	2.16.840.1.113883.5.55	M	0	A
10011	2.16.840.1.113883.5.55	R	0	A
10013	2.16.840.1.113883.5.55	RQ	0	A
10014	2.16.840.1.113883.5.55	NP	0	A
10015	2.16.840.1.113883.5.55	NR	0	A
10016	2.16.840.1.113883.5.55	RE	0	A
10017	2.16.840.1.113883.5.55	X	0	A
10019	2.16.840.1.113883.5.57	R	0	A
10020	2.16.840.1.113883.5.57	D	0	A

10021	2.16.840.1.113883.5.57	I	0	A
10022	2.16.840.1.113883.5.57	K	0	A
10023	2.16.840.1.113883.5.57	V	0	A
10025	2.16.840.1.113883.5.57	ESA	0	A
10026	2.16.840.1.113883.5.57	ESD	0	A
10027	2.16.840.1.113883.5.57	ESC	0	A
10028	2.16.840.1.113883.5.57	ESAC	0	A

Table: VCS_concept_designation

Internal Id	Designation	seq - for case differences	language	preferredForLanguage
10011	Mandatory	0	en	-1
10011	Required - V2.x	0	en	0

Query of HL7 internal id, concept code and designation:

codeSystemName	Code system OID	Internal concept identifier	Concept code	Designation
HL7ConformanceInclusion	2.16.840.1.113883.5.55	10011	R	Required - V2.x
HL7ConformanceInclusion	2.16.840.1.113883.5.55	10011	M	Mandatory
HL7ConformanceInclusion	2.16.840.1.113883.5.55	10011	M	Required - V2.x
HL7ConformanceInclusion	2.16.840.1.113883.5.55	10011	R	Mandatory

To represent HL7 in LexGrid:

A single coding scheme will be created in LexGrid.

Each **VCS_concept_code_xref.internalId** will be represented as a LexGrid Concept Code.

The LexGrid Concept Code will be generated by the concatination of **VCS_concept_code_xref.internalId** and **VCS_concept_code_xref.conceptCode2** (separated by a colon ':').

Not only the duplicates that exist within coding schemes will be dealt with using the id/mnemonic concatenation but also those duplicates that exist between coding schemes.

A LexGrid Concept Code Presentation Property will be created for each HL7 designation (VCS_concept_designation).

The Presentation Property will include Presentation (HL7 Designation), Source (HL7 codeSystemName) and a Property Qualifier of source-code (HL7 Concept Code).

For example, the following structure represents both HL7 10011 entries in code system

2.16.840.1.113883.5.55:

Coding Scheme: HL7 - urn:oid:2.16.840.1.113883.3.26.1.2

Concept Code: 10011:M

Entity Description: >The message element must appear every time the message is communicated and its value must not be null. This condition is subject to the rules of multiplicity and conditionality. If a non-null default value is defined for the element, a null value may be communicated.

Status: Active

Is Active: true

Is Anonymous: false

Presentation: Mandatory

> **Property Name:** textualPresentation
>
> **Property Id:** T-1
>
> **Language:** ENG
>
> **Is Preferred:** true
>
> **Representational Form:** PT
>
> **Source:** HL7ConformanceInclusion , **Role:** null, **SubRef:** null
>
> **Property Qualifier Id:** source-code , **Property Qualifier Content:** M

Presentation: Required - V2.x

> **Property Name:** textualPresentation
>
> **Property Id:** T-2
>
> **Language:** ENG
>
> **Is Preferred:** false
>
> **Representational Form:** PT
>
> **Source:** HL7ConformanceInclusion, **Role:** null, **SubRef:** null
>
> **Property Qualifier Id:** source-code , **Property Qualifier Content:** M

Coding Scheme: HL7 - urn:oid:2.16.840.1.113883.3.26.1.2

Concept Code: 10011:R

Entity Description: >The message element must appear every time the message is communicated and its value must not be null. This condition is subject to the rules of multiplicity and conditionality. If a non-null default value is defined for the element, a null value may be communicated.

Status: Active

Is Active: true

Is Anonymous: false

Presentation: Mandatory

> **Property Name:** textualPresentation
>
> **Property Id:** T-1
>
> **Language:** ENG
>
> **Is Preferred:** true
>
> **Representational Form:** PT
>
> **Source:** HL7ConformanceInclusion , **Role:** null, **SubRef:** null
>
> **Property Qualifier Id:** source-code , **Property Qualifier Content:** R

Presentation: Required - V2.x

> **Property Name:** textualPresentation
>
> **Property Id:** T-2
>
> **Language:** ENG
>
> **Is Preferred:** false
>
> **Representational Form:** PT
>
> **Source:** HL7ConformanceInclusion, **Role:** null, **SubRef:** null
>
> **Property Qualifier Id:** source-code , **Property Qualifier Content:** R

Loading the HL7 Rim as a monolithic coding scheme

1. Load coding scheme data as HL7 Rim Metadata from the Model table (rather than the coding scheme data for each HL7 coding scheme).
    a. Mapping of these values will be incomplete:

i. Mapping proposal:

LexGrid	HL7 RIM
<codingSchemeName>	<modelID>
<formalName>	<name>
<registeredName>	http://www.hl7.org/Library/data-model/RIM *
<defaultLanguage>	en*
<representsVersion>	<versionNumber>
<isNative>	0*
<approximateNumberofConcepts>	Result of count on concept bearing table?
<firstRelease>	MISSING
<modifiedInRelease>	MISSING
<deprecated>	MISSING
<entityDescription>	<description>
<copyright>	MISSING

b. No URN exists and we may need to consider creating one (see entry for registeredName).

2. Locate and load all mappings (such as supportedAssociations and supportedProperties).
   a. Create a supportedHiearchy with a root node of @ on hasSubtype?

3. Iterate through the code system table rows and get each coding scheme.
   a. Create and persist an “@” node in the database
   b. Prepare an artificial “top node“ for each coding scheme. (Metadata persisted here as concept properties?) This will result in 250 top nodes.
      i. The artificial top nodes will need to have a concept code created for them.
      ii. Attach to “@” the artificial top nodes as a hasSubtype.
      iii. Locate the actual top nodes of each coding scheme by querying the relations table to see if they exist as a target code, if not, they are top nodes so attach them to the artificial top node via hasSubtype.
   c. Translate the RRF source property loads to the EMF world.
      i. Load the concepts ensuring that the coding scheme name is loaded as a “source” property
      ii. Load the relations ensuring that the source and target coding scheme data is loaded with the coding scheme name.
4. Concurrent to this process create an updated “HL7 RIM to LexGrid for NCI” mapping from the current Excel mapping document.

LexGrid Text

The text files that can be imported must use the following formats. Items surrounded by <> are required. Items further surrounded by [] are optional. \t represents a tab - the default delimiter - however other delimiters may be used.

Lines beginning with # are comments.

Format A:

```
<codingSchemeName>\t<codingSchemeId>\t<defaultLanguage>\t<formalName>[\t<version>][\t<source>][\t<desc
<name1>[\t <description>]
\t <name2>[\t <description>]
\t\t <name3>[\t <description>]
```

The leading tabs represent hierarchical “hasSubtype” relationship nesting :

(name1 hasSubtype name2 and name2 hasSubtype name3)

Concept Codes will be automatically generated.

If a name is used more than once - it will be assigned the same code.

If a description is used more than once (for a given name) only the first description will be stored.

Format B:

In this format, concept codes can be provided. This is the same as “Format A” with the inclusion of concept codes as part of the input.

```
<code>\t<name>[\t<description>]
```

If the same code occurs twice, the names must match. Description rules same as “Format A.”

Example of Format A

#lines starting with "#" are comments

#blank lines are ok

#the first "real" line of the file must be of the following format:

#<codingSchemeName>\t<codingSchemeId>\t<defaultLanguage>\t<formalName>[\t<version>][\t

```
][\t<description>][\t<copyright>]

colors    1.2.3     en        colors coding scheme 1.0      Someone’s Head      a simple example codi

<nowiki>#</nowiki>The rest of the file (for format A) should look like this:

<pre>ColorHolder of colors
          Red
          Green     The color Green
                    Light Green         foobar
                    Dark GreenThe color dark green
          Blue
                    Red
                              Green     The color Green</pre>

Example of Format B

<nowiki>#</nowiki>lines starting with "#" are comments

<nowiki>#</nowiki>blank lines are ok

<nowiki>#</nowiki>the first "real" line of the file must be of the following format:
<nowiki>#</nowiki><codingSchemeName>\t<codingSchemeId>\t<defaultLanguage>\t<formalName>[\t<version>][

colors2   1.2.4     en        colors coding scheme1.1       Someone’s Head      a simple example codi

<nowiki>#</nowiki>The rest of the file (for format B) should look like this:

<pre>1    Color     Holder of colors
          4         Red
          6         Green     The color Green
                    7         Light Green
                    8         Dark Green
          5         Blue
                    8         Dark GreenThe color dark green
```

Retrieved from "https://cabig-kc.nci.nih.gov/Vocab/KC/index.php/LexEVS_5.x_Loader_Source_Mapping"
Categories: VKC Contents | Documentation | LexGrid

- This page was last modified on 31 January 2010, at 03:38.

CONTACT US PRIVACY NOTICE DISCLAIMER ACCESSIBILITY APPLICATION SUPPORT

National Cancer Institute
U.S. National Institutes of Health | www.cancer.gov
caBIG Knowledge Center
A part of the Enterprise Support Network

- Home
- Knowledge Centers
  - caGrid
  - Clinical Trials Management Systems
  - Data Sharing and Intellectual Capital
  - Molecular Analysis Tools
  - Tissue/Biospecimen Banking and Technology Tool
  - Vocabulary
- Discussion Forums
  - caBIG General Forum
  - caGrid
  - Clinical Trials Management Systems
  - Data Sharing and Intellectual Capital
  - Molecular Analysis Tools
  - Tissue/Biospecimen Banking and Technology Tool
  - Vocabulary
- Bugs/Feature Requests
- Development Code Repository

LexEVS 5.x Loader Framework

From Vocab_Wiki

LexEVS 5.x Loader Source Mapping > Main Page > LexEVS 5.x Loader Guide > Main Page > LexEVS 5.x Loader Framework

Contents

Introduction

This document is a section of the Loader Guide. It is new in LexEVS v5.1.

Document Purpose

This document provides the detailed design and implementation of the Loader Framework Extension. It is the goal of this document to provide enough information to enable application developers to create custom loaders. This document assumes the developer is already familiar with the LexEVS software.

Loader Framework Background and Enhancements

Previous versions of LexEVS software has provided a set of loaders within an existing legacy framework which served LexEVS developers well over many years. But as LexEVS has gained users, and requests for new loaders have grown, it was decided to create a new loader framework. Specifically, this development work addresses "TASK 6 - IMPROVE LEXEVS LOADING FRAMEWORK" in the National Cancer Institute (NCI) Statement of Work (SOW) document (reference ?????).

The LexEVS v5.1 loader framework meets these emerging needs compared to the loader framework of previous versions:

- is easier to extend
- provides improved performance
- enables dynamic loading of new loaders
- leverages proven open source components, such as Spring Batch and Hibernate

Also, the new framework is completely independent of existing loader code, so there is no impact to existing loaders.

Scope

The LexEVS v5.1 Loader Framework provides a way for LexEVS developers to write new loaders and have them recognized dynamically by the LexEVS code. Also the framework provides help to loader developers in the form of utility classes and interfaces.

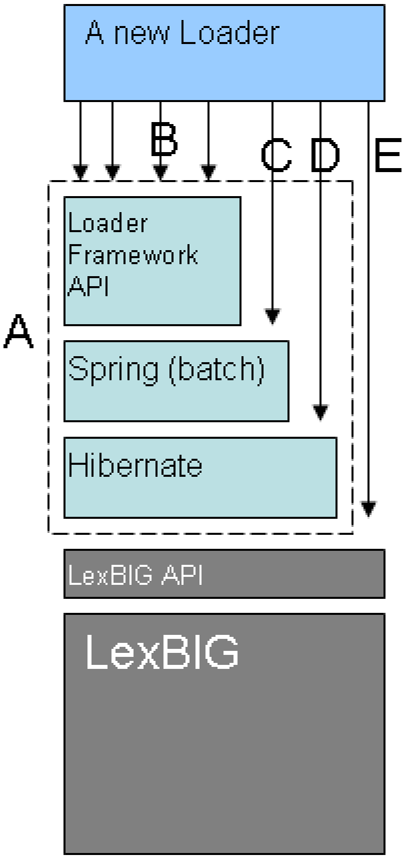
Architecture

The LexEVS v5.1 Loader Framework extends the functionality of LexEVS 5.0. For more information on LexBIG, see LexEVS_Version_5.0.

The image below shows the major components of the Loader Framework.

(A) A hypothetical new loader in relation to the loader framework, and what expected API usage would be.
(B) Ideally, the new loader can make most if its API calls through the utilities provided by the Loader Framework API.
(C) Some work will need to be done with Spring (C) such as configuration of a Spring config file.
(D and E) It may or may not be necessary for a loader to use Hibernate or the LexBIG API. Again, the hope is that much of the work a new loader may need to do can be accomplished by the Loader Framework API.

The Loader Framework utilizes Spring Batch for managing its Java objects to improve performance and Hibernate provides the mapping to the LexGrid database.



Assumptions

None

Dependencies

- This Loader Framework requires LexEVS release 5.0 or above.
- Development systems are required to install the Sun Java Development Kit (SDK) or Java Runtime Environment (JRE) version 1.5.0_11 or above.
- Maven 2.1 or greater.
- For software and hardware dependencies for the system hosting the LexEVS runtime, refer to Installation and downloads.

Issues

None

Development and Build Environment

Third Party Tools

- Spring: A lightweight open-source application framework.
  - Spring: see [1]
  - Spring Batch: see [2]
  - Sprint Batch Reference: see [3]
- Hibernate: An open source Java persistence framework; see [4]
- Maven: Apache build manager for Java projects; see [5]
- Eclipse: An Open Source IDE. See [6]

Loader Framework Code

The Loader Framework code is available in the NCI Subversion (SVN) repository. It is comprised of three Framework projects. Also at the time of this writing there are three projects in the repository that utilize the Loader Framework.

Loader Framework Projects

- PersistanceLayer: a Hibernate connector to the LexBIG database
- Loader-framework: a framework that sets up build information for Maven
- Loader-framework-core: a framework that contains all the interfaces and utilities; also contains an extendable class "AbstractSpringBatchLoader" that all new Loaders should extend

Loader Proejcts Using the New Framework

- abstract-rrf-loader: a holder for common rrf-based loader code
- meta-loader: a new loader to read the NCI MetaThesaurus
- umls-loader: a loader for reading Unified Medical Language System (UMLS) content

Maven

The above projects utilize Maven for build and dependency management. Obtain the Maven plugin for Eclipse at [7]

How to Use the Loader Framework: A Roadmap

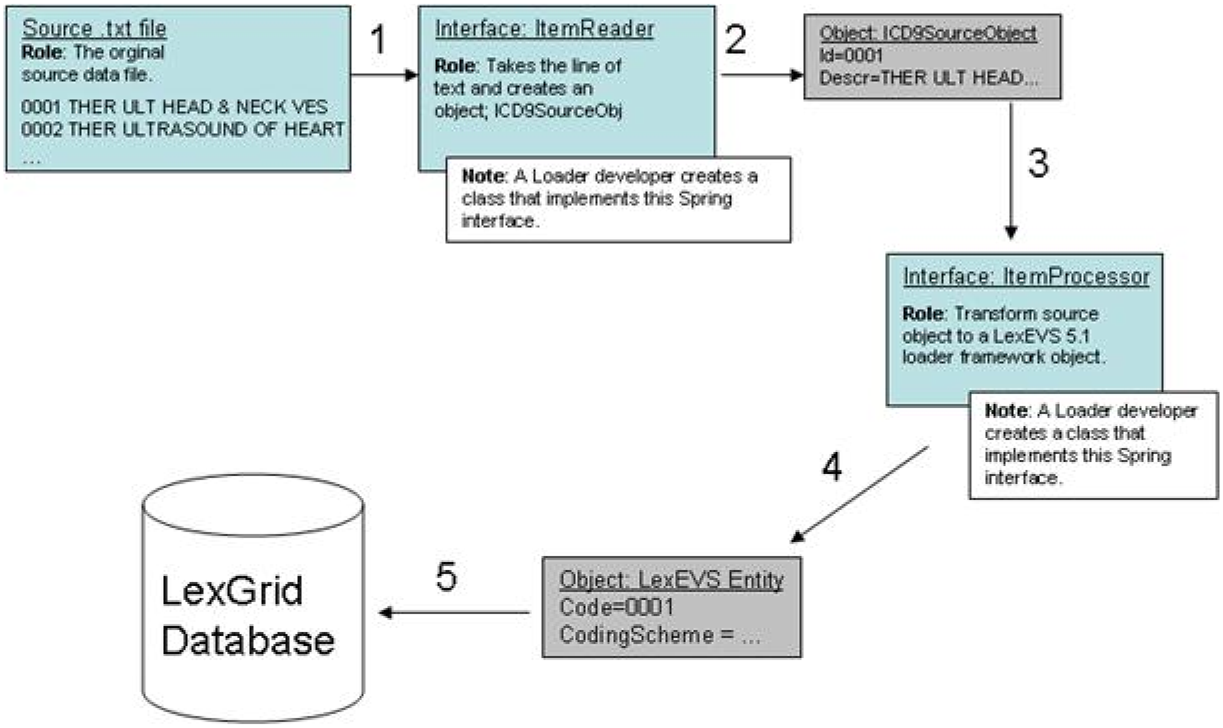
You can write a loader that uses the Loader Framework. The loader would follow this general process:

1. Read the raw data from the file into intermediate data structures, such as a user-defined ICD9SourceObject object.
2. Process the user-defined objects into LexGrid model objects.
3. Write the data in the LexGrid objects to the database.

An example may help in understanding the Framework. Our discussion will refer to the illustration below. Let's say we are writing a loader to load the ICD-9-CM codes and their descriptions, which are contained in a text file. We know we'll need a data structure to hold the data after we've read it so we have a class:

```
ICD9SourceObject {
String id;
String descr;
String getId() { return id; }
}
```

The Loader Framework uses Spring Batch to manage the reading, processing, and writing of data. Spring provides classes and interfaces to help do this work, and the Loader Framework also provides utilities to help loader developers. In our example, illustrated below, we will write a class that will use the Spring ItemReader interface. It will take a line of text and return an ICD9SourceObject (shown as 1 and 2). Next we'll want to process that data into a LexEVS object such as an Entity object. So we'll write class that implements Spring's ItemProcessor interface. It will take our ICD9SourceObject and output a LexEVS Entity object (shown as 3 and 4). Finally, we'll want to write the data to the database (shown as 5). Note that the LexEVS model objects provided in the Loader Framework are generated by Hibernate and utilize Hibernate to write the data to the database. This will free us from having to write SQL.


Source .txt file
Role: The orginal source data file.
0001 THER ULT HEAD & NECK VES
0002 THER ULTRASOUND OF HEART
...
1
Interface: ItemReader
Role: Takes the line of text and creates an object; ICD9SourceObj
Note: A Loader developer creates a class that implements this Spring interface.
2
Object: ICD9SourceObject
Id=0001
Descr=THER ULT HEAD...
3
Interface: ItemProcessor
Role: Transform source object to a LexEVS 5.1 loader framework object.
Note: A Loader developer creates a class that implements this Spring interface.
4
Object: LexEVS Entity
Code=0001
CodingScheme = ...
5
LexGrid Database

Spring

Configure Spring to be aware of your objects and to manage them. This is done via an XML configuration file. More details on the Spring config file arebelow.

ItemReader/ItemProcessor

Either write a class implementing this interface or use one of the Spring helper classes that already implement this interface. If you use one of the Spring classes, you may need to provide one of your own helper classes to construct your internal data structure object, such as ICD9SourceObject. Provide it to the Spring object via a setProperty call configured in the Spring config file.

Maven Set up

The projects containing the Loader Framework (**PersistanceLayer** , **loader-framework** , and **loader-framework-core**) use Maven for dependency management and build. You will still use Eclipse as your IDE and code repository, but you will need to install a Maven plugin for Eclipse.

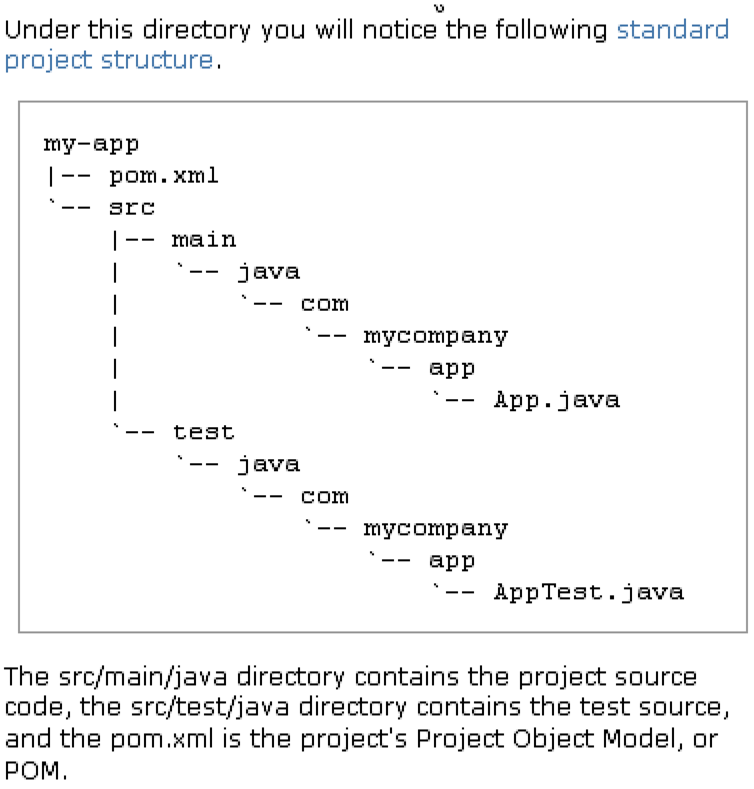
1. Install the Maven plugin for Eclipse, which can be found at: [8].
2. Provide a URL and userid/password to a Maven repository on a server (which manages your dependencies or dependent jar files). Ours here at Mayo is: [9].
3. Import the Loader Framework classes from SVN.
4. You will most likely see build errors about missing jars. Resolve those by right clicking on the project with errors, select **Maven'**, and **Resolve Dependencies**. This will pull the dependant jars from the Maven repository into your local environment.
5. To build a Maven project, right click on the project, select **Maven**, then select

assembly:assembly.

Eclipse Project Set up

When you create a new loader project in Eclipse, it is recommended you follow the Maven directory structure. By following this convention, Maven can build the project and find the test cases.

The following diagram is from the Maven documentation:



Under this directory you will notice the following standard project structure.

```
my-app
|-- pom.xml
`-- src
    |-- main
    |   `-- java
    |       `-- com
    |           `-- mycompany
    |               `-- app
    |                   `-- App.java
    `-- test
        `-- java
            `-- com
                `-- mycompany
                    `-- app
                        `-- AppTest.java
```

The src/main/java directory contains the project source code, the src/test/java directory contains the test source, and the pom.xml is the project's Project Object Model, or POM.

For more information on the Maven project, see [10]

Configure your Spring Config (myLoader.xml)

Spring is a lightweight bean management container; among other things, it contains a batch function that is utilized by the Loader Framework. A loader using the framework will need to work closely with Spring Batch. The way it does that is through Spring's configuration file where you configure beans (your loader code) and how the loader code should be utilized by Spring Batch (by configuring a Job, Step, and other Spring Batch stuff in the spring config file). Here is sample code:

```
<job id="ioSampleJob">
   <step name="step1">
      <tasklet
         <chunk reader="fooReader" processor="fooProcessor" writer="compositeItemWriter" commit-interv
         </chunk>
      </tasklet>
   </step>
</job>

<bean id= "compositeItemWriter" class="...compositeItemWriter">
   <property name="delegate" ref="barWriter" />
</bean>

<bean id="barWriter" class="...barWriter" />
```

What follows is a brief overview of those tags related to the LoaderFramework. For more detail please see the Spring documentation at [11].

Beans

The **beans:beans** tag is the all-encompassing tag. You define all your other tags in it. You can also define an import within this tag to import an external Spring config file. (Import is not shown in the sample image above.)

Bean

Use these tags, **beans:bean'**, to define the beans to be managed by the Spring container by specifying the packaged qualified class name. You can also specify inititialization values and set bean properties within these tags.

```
<beans:bean id="umlsCuiPropertyProcessor" parent="umlsDefaultPropertyProcessor" class="org.lexgrid.loa
  <beans:property name="propertyResolver" ref="umlsCuiPropertyResolver" />
</beans:bean>
```

Job

The **job** tag is the main unit of work. The job is comprised of one or more steps that define the work to be done. Other advanced and interesting things can be done within the Job such as using **split** and **flow** tags to indicate work that can be done in parellel steps to improve performance.

```
<job id="umlsJob" restartable="true">
 <step id="populateStagingTable" next="loadHardcodedValues" parent="stagingTablePopulatorStepFactory"/>
...
```

Step

One or more **step** tags make up a job and can vary from simple to complex in content. Among other things, you can specify which step should be executed next.

Tasklet

You can do anything you want within a Tasklet, such as sending an email or a LexBIG function such as indexing. You are not limited to just database operations. The Spring documentation also has this to say about Tasklets:

```
The Tasklet is a simple interface that has one method, execute, which will be a called repeatedly
by the TaskletStep until it either returns RepeatStatus.FINISHED or throws an exception to signal
a failure. Each call to the Tasklet is wrapped in a transaction.
```

Chunk

Spring documentation says it best:

```
Spring Batch uses a "Chunk-Oriented" processing style within its most common implementation. Chunk-
oriented processing refers to reading the data one at a time, and creating "chunks" that will be
written out, within a transaction boundary. One item is read in from an ItemReader, handed to an
ItemWriter, and aggregated. Once the number of items read equals the commit interval, the entire
chunk is written out via the ItemWriter, and then the transaction is committed.
```

Reader

An attribute of the **chunk** tag. Here is the class that you defined implementing the Spring ItemReader interface to read data from your data file and create domain-specific objects.

Processor

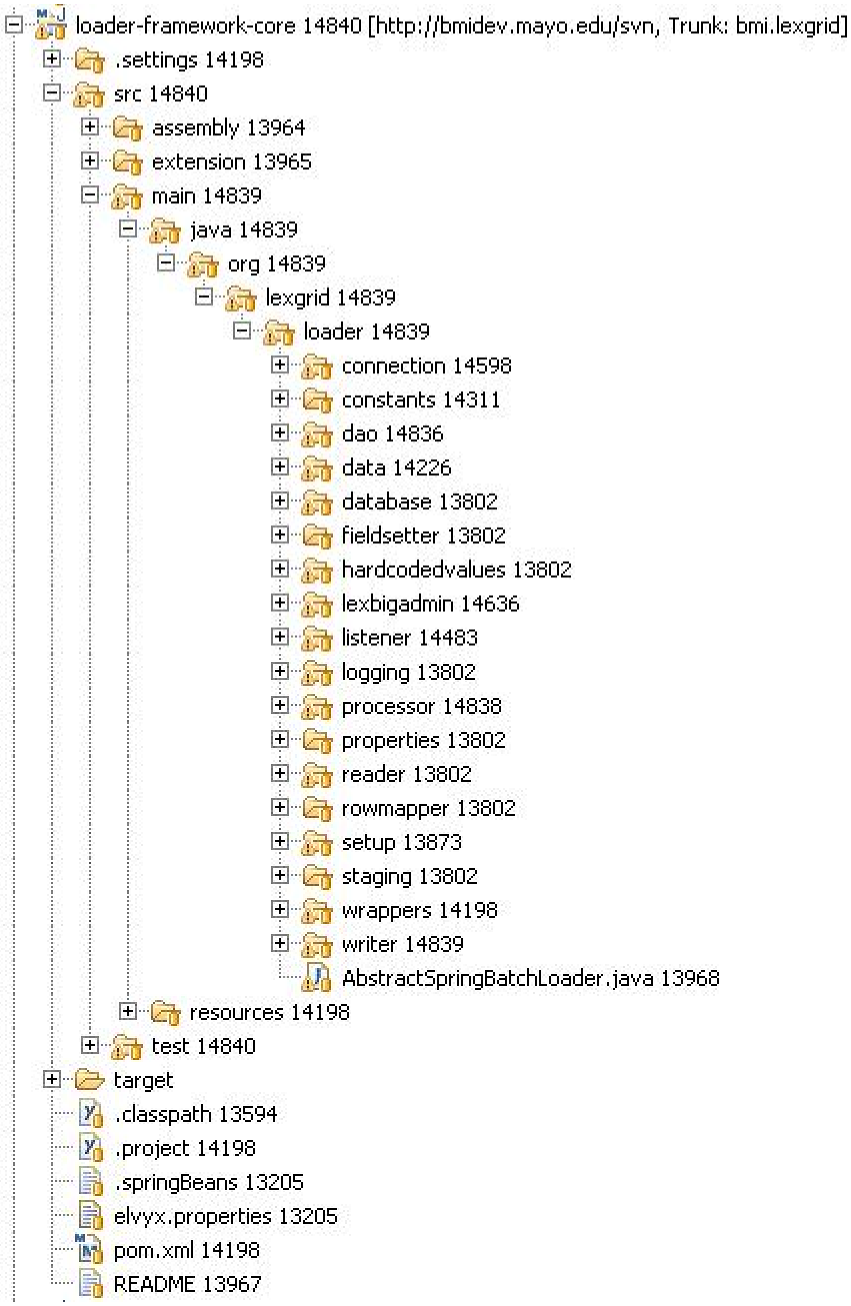
Another attribute of the **chunk** tag. This is the class that implements the ItemProcessor interface where other manipulations of the domain objects take place. In the case of the Loader Framework, we create LexGrid model objects from the domain objects so that they can be written to the database via Hibernate. Note that this is not a required attribute. In theory, if you had a data source from which you could read such that you could create LexBIG objects immediately, you would not need a processor. In practice this would most likely not be the case, but rather you need to work with the data to get it into LexBIG objects.

Writer

Attribute of the **chunk** tag. This class will implement the Spring interface ItemWriter. In the case of the Loader Framework, these classes have been written for you. They are the LexGrid model objects that use Hibernate to write to the database.

Key Directories

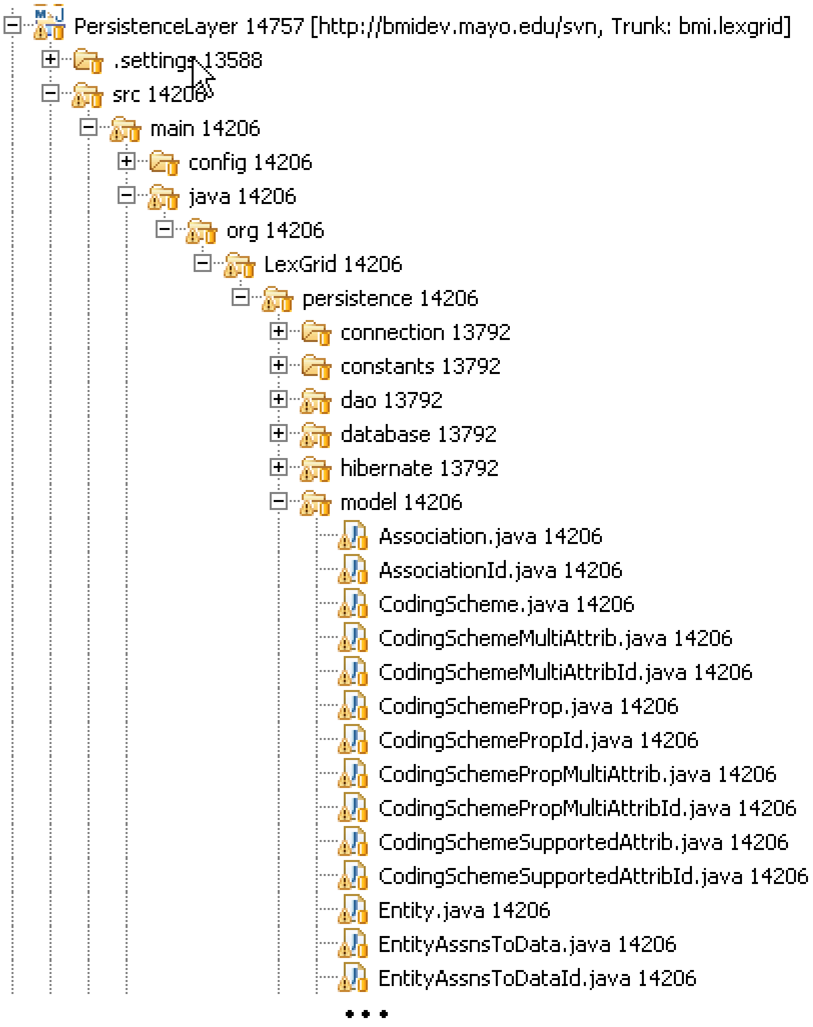
Below is an image of the loader-framework-core project in Eclipse, which shows the key directories of the Loader Framework. The following is a summary of the contents of those directories.



```
loader-framework-core 14840 [http://bmidev.mayo.edu/svn, Trunk: bmi.lexgrid]
    .settings 14198
    src 14840
        assembly 13964
        extension 13965
        main 14839
            java 14839
                org 14839
                    lexgrid 14839
                        loader 14839
                            connection 14598
                            constants 14311
                            dao 14836
                            data 14226
                            database 13802
                            fieldsetter 13802
                            hardcodedvalues 13802
                            lexbigadmin 14636
                            listener 14483
                            logging 13802
                            processor 14838
                            properties 13802
                            reader 13802
                            rowmapper 13802
                            setup 13873
                            staging 13802
                            wrappers 14198
                            writer 14839
                            AbstractSpringBatchLoader.java 13968
            resources 14198
        test 14840
    target
    .classpath 13594
    .project 14198
    .springBeans 13205
    elvyx.properties 13205
    pom.xml 14198
    README 13967
```

Directory	Summary
connection	Connect to LexBIG and do LexBIG tasks such as register and activate
constants	Assorted constants
dao	Access to the LexBIG database
data	Directly related to data going into the LexBIG database tables
database	Database-specific tasks not related to data, such as finding out the database type (MySQL, Oracle)
fieldsetter	Spring-related classes for helping to write to the database
lexbigadmin	Common tasks for LexBIG to perform, such as indexing

listener	Listeners you can attach to a load so that the code will execute at certain points in the load, such as a cleanup listener that runs when the load is finished, or a setup listener, etc.
logging	Access to the LexBIG logger
processor	*Important directory:* classes to which you can pass a domain-specific object and which will return a LexBIG object
properties	Code used internally by the Loader Framework
reader	Readers and reader-related tools for loader developers
rowmapper	Classes for reading from a database; currently experimental code
setup	*Loader developers should not need to dive into this directory*. Classes such as JobRepositoryManager that help Spring do its work; as Spring hums along it keeps tables of its internal workings.
staging	Helper classes to use if your loader needs to load data to the database temporarlily
wrappers	Helper classes and data strucutres such as a Code/CodingScheme class
writer	Miscellanous classes that write to the database. These are not the same classes you would use in your loader, i.e the LexBIG model objects that use Hibernate. Those classes are contained in the PersistanceLayer project (shown below). It is by using those classes in the PersistenceLayer that you let the Loader Framework do some of the heavy lifting for you.



```
PersistenceLayer 14757 [http://bmidev.mayo.edu/svn, Trunk: bmi.lexgrid]
  .settings 13588
  src 14206
    main 14206
      config 14206
      java 14206
        org 14206
          LexGrid 14206
            persistence 14206
              connection 13792
              constants 13792
              dao 13792
              database 13792
              hibernate 13792
              model 14206
                Association.java 14206
                AssociationId.java 14206
                CodingScheme.java 14206
                CodingSchemeMultiAttrib.java 14206
                CodingSchemeMultiAttribId.java 14206
                CodingSchemeProp.java 14206
                CodingSchemePropId.java 14206
                CodingSchemePropMultiAttrib.java 14206
                CodingSchemePropMultiAttribId.java 14206
                CodingSchemeSupportedAttrib.java 14206
                CodingSchemeSupportedAttribId.java 14206
                Entity.java 14206
                EntityAssnsToData.java 14206
                EntityAssnsToDataId.java 14206
                 • • •
```

Algorithms

None

Batch Processes

None

Error Handling

Spring Batch gives the Loader Framework some degree of recovery from errors. Like the other features of Spring, error handling is something you need to configure in the Spring config file. Basically, Spring will keep track of the steps it has executed and make note of any step that has failed. Those failed steps can be re-run at a later time. The Spring documentation provides additional information on this function. See [12] and [13].

Database Changes

None

Client

Loaders written to use the new framework will be called via the command line or script. Currently, the LexBIG GUI does not provide a framework to dynamically load extendable GUI components.

JSP/HTML

None

Servlet

None

Security Issues

None

Performance

Spring can accommodate parallel processing to enhance performance. The Spring documentation provides a good discussion of this topic. See [14].

Internationalization

Not internationalized

Installation / Packaging

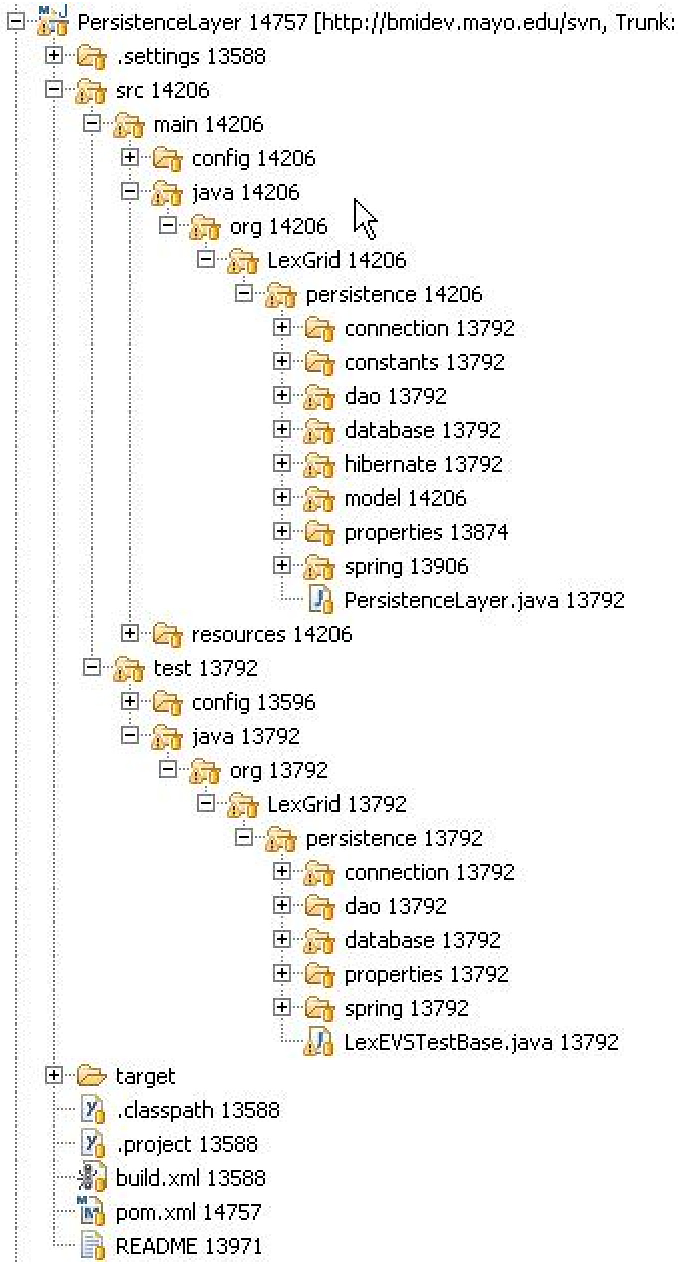
The Loader Framework is packaged as a LexBIG extension and thus is not included in the LexBIG jar

Migration

None

Testing

Automated tests are run via Maven. As mentioned earlier, the projects containing the Loader Framework code are configured to work with Maven. The illustration below shows the PersistenceLayer project and its standard Maven layout. Notice the structure of the test code mirrors the structure of the application code. To run the automated test in our Eclipse environment, we select the project, right click, select **Run As** and select **Maven test**. Maven does the rest.



```
PersistenceLayer 14757 [http://bmidev.mayo.edu/svn, Trunk:
  .settings 13588
  src 14206
    main 14206
      config 14206
      java 14206
        org 14206
          LexGrid 14206
            persistence 14206
              connection 13792
              constants 13792
              dao 13792
              database 13792
              hibernate 13792
              model 14206
              properties 13874
              spring 13906
              PersistenceLayer.java 13792
      resources 14206
    test 13792
      config 13596
      java 13792
        org 13792
          LexGrid 13792
            persistence 13792
              connection 13792
              dao 13792
              database 13792
              properties 13792
              spring 13792
              LexEVSTestBase.java 13792
  target
  .classpath 13588
  .project 13588
  build.xml 13588
  pom.xml 14757
  README 13971
```

Test Guidelines

The test cases are also integrated into the LexBIG 5.1 build environment and are run with each build.

Test Cases

See System Testing

Test Results

See System Testing

Custom Loader Feasibility Report and Recommendation

Persistence Layer Feasibility

The Persistence Layer enables LexEVS to have a single access point to the underlying database. This has several advantages:

- The DAO is implemented as an Interface, not a concrete class. We can implement this interface with Hibernate, JDBC, Ibatis, or any other Persistence tool or framework.
- All loaders can now share a single entry point to the database, and are not limited by memory constraints as some of the EMF persistence was.
- Connection Pooling and management is abstracted from the code and pluggable. Data source implementations may be switched and Connection Pooling may be configured without recompiling code.
- Transactions may be defined programmatically via AOP interceptors.

As LexEVS moves forward, the Persistence Layer is also flexible enough to play a part in the runtime Query API. With this, the runtime and loader code would be able to share a common Data Access Layer - we would then have a true DAO Layer.

Loader Framework Feasibilty

The Loader Framework has been implemented for two loaders: the UMLS single ontology loader and the NCI Metathesaurus loader. These loaders that implement the Loader Framework simply must define the READ and TRANSFORMATION mechanisms for the load, as well as load order and flow. All common details of loading to LexEVS will be dealt with by the Loader Framework and will not have to be implemented. Tools exist for:

- Lucene indexing
- registering CodingSchemes
- changing CodingScheme status (to ACTIVE, INACTIVE, etc)
- building the Transitivity Closure table
- adding supported attributes
- detecting database type
- staging temporary data to the database
- restarting failed loads
- integrating with LexEVS logging
- detecting and handling root nodes
- additional common LexEVS load-related tasks

Also, to aid in transformation, basic building blocks have been created that users may extend, such as:

- processors for all of LexEVS Model Objects
- various list processors
- grouping processors
- auto-supported attribute adding processors
- several basic resolvers to extract LexEVS Specific data from the source
- various other processors for specialized tasks

Several Utilities are also available for reading and writing, such as:

- group readers
- group writers
- writers configurable to skip certain records
- partitionable readers to break up large source files
- error-checking readers and Writers
- a validating framework for inspecting content before it is inserted into the database

Retrieved from "https://cabig-kc.nci.nih.gov/Vocab/KC/index.php/LexEVS_5.x_Loader_Framework"
Categories: VKC Contents | Documentation | LexEVS

- This page was last modified on 22 December 2009, at 13:00.