

LexEVS 6.0.0 Release Notes

Developers:

Craig Stancl; Deepak Sharma; Kevin Peterson; Scott Bauer; Sridhar Dwarkanath; Zonghui Lian;
Division of Biomedical Statistics and Informatics,
Department of Health Science Research
Mayo Foundation, Rochester, MN

Table of Contents

- [Introduction](#)
- [CTS2 Implementation](#)
- [OWL/RDF Export](#)
- [Integer Primary Keys](#)
- [Mapping Extension](#)
- [Supplement Extension](#)
- [Value Sets \(gForge # 26650\)](#)
- [Versioning \(gForge # 26651\)](#)
- [XML Loaders](#)
- [XML Exporter](#)
- [GForge Items](#)
- [LexGrid model changes](#)

Introduction

This document contains major functionalities that are included in LexEVS 6.0.0 Release.

CTS2 Implementation

- Initial Interfaces
 - The initial set of Interface and API has been created. Input parameters and output values are still evolving, but interfaces are in place reflect the functionality of CTS2.

CTS2 Administration Development (see: [org.lexevs.cts2.admin.AdminOperation](#))

- Import Code System
 - (see: [org.lexevs.cts2.admin.load.CodeSystemLoadOperation](#))
 - Loads a 'NEW' Revision of a Coding Scheme into the System. If Revision info is not supplied by the user (such as Revision Id), it will be generated. Utilizes existing LexEVS loader functionality.
- Change Code System Status
 - (see: [org.lexevs.cts2.author.CodeSystemAuthoringOperation](#))
- Import Value Set Version
 - (see: [org.lexevs.cts2.admin.load.ValueSetLoadOperation](#))
- Export Code System Content
 - (see: [org.lexevs.cts2.admin.load.CodeSystemLoadOperation](#))
- Register for Notification
 - (see: [org.lexevs.cts2.admin.NotificationAdminOperation](#))
 - Allows the user to register for notification of a System State Change. There are various events that occur in the system, and a user may choose to listen for one or more of them.
- Update Notification Registration
 - (see: [org.lexevs.cts2.admin.NotificationAdminOperation](#))
 - Allows the user to alter an existing Notification Registration.
- Update Notification Registration Status
 - (see: [org.lexevs.cts2.admin.NotificationAdminOperation](#))
 - Allows the user to 'SUSPEND', 'REINSTATE', or 'REMOVE' a notification from the system.

CTS2 Authoring Development (see: [org.lexevs.cts2.author.AuthoringOperation](#))

- Association Authoring Operations (see: [org.lexevs.cts2.author.AssociationAuthoringOperation](#))
 - Create association in an established coding scheme using the Revision model.
 - Create association in an established mapping dedicated coding scheme using the Revision model.
 - Create association mapping. Create mappings between coding schemes and persist as new coding scheme with default metadata settings. Persist using the Revision model.
 - Create association mapping. Create mappings between coding schemes and persist as a coding scheme with user defined metadata values. Persist using the Revision model.
 - Create association mapping: Create mappings between coding schemes with persisted concept values pulled from target schemes current in a LexEVS service instance. Persist using the Revision model.
 - Create association type. Create a new AssociationPredicate value and persist is using the Revision model.
- ValueSet Authoring
 - (see: [org.lexevs.cts2.author.ValueSetAuthoringOperation](#))

OWL/RDF Export

- OWL/RDF Export functionality has been added, allowing the export of loaded LexEVS content into OWL format.
 - (see: [org.LexGrid.LexBIG.Extensions.Export.OWL_Exporter](#))
- **Known issues/restrictions:**
 - GForge #30058: AssociationData is not exported. Thus, the owl/rdf exporter cannot handle the owl:hasValue, owl:maxCardinality, owl:minCardinality, owl:cardinality constraints.

Integer Primary Keys

- Primary keys are now configurable – either a GUID or a Sequential Integer scheme can be used. Sequential Integers offer better performance, while GUIDs enable portability between databases.

The configuration will appear as such in the lbconfig.props file:

1. DB_PRIMARY_KEY_STRATEGY indicates which strategy will be used
2. for the primary key of the database tables.
3. WARNING - This cannot be change after the initial
4. schema installation.
#
5. Allowable values include:

#"GUID"
 - Primary Keys are implemented as random GUIDs.
#"SEQUENTIAL_INTEGER"
 - Primary Keys will be sequentially incremented
 - as Integer values.
DB_PRIMARY_KEY_STRATEGY=GUID

Mapping Extension

(see: [org.LexGrid.LexBIG.Extensions.Generic.MappingExtension](#))

- A general purpose Generic Extension for Mapping Coding Schemes has been introduced.

The main implemented features are:

- Faster retrieval of Mapping ontology relationships as compared to standard CodedNodeGraph methods
- Expanded Sorting capabilities to include:
 - Source Code
 - Target Code
 - Source Entity Description
 - Target Entity Description
 - Relationship Name
 - A Named Qualifier
- Ability to count the number of codes that participate in a mapping ontology without having to traverse
- Determine whether or not an Ontology can be considered a 'Mapping' Ontology.
 - A 'Mapping' Ontology is defined in this case as having at least ALL Relations containers marked as 'isMapping=true'

Supplement Extension

(see: [org.LexGrid.LexBIG.Extensions.Generic.SupplementExtension](#))

- A general purpose Generic Extension for Coding Scheme Supplements has been introduced.

The main implemented features are:

- Determine whether or not a given Coding Scheme is acting as a Coding Scheme Supplement.
- Resolve the Parent Coding Scheme of a given Coding Scheme Supplement.

Value Sets (gForge # 26650)

- Administrative functions :
 - Ability to load Value Set and Pick List Definitions in to LexGrid repository
 - Ability to remove Value Set and Pick List Definitions from LexGrid repository.
 - Ability to export Value Set and Pick List Definition in LexGrid XML format.
- Query functions :
 - Ability to list all the Value Set and Pick List Definitions loaded in the system.
 - Ability to dynamically resolve Value Set and Pick List Definition with/without user supplied restrictions.
 - Ability to check if a concept code is part of given Value Set.
 - Ability to check if one Value Set is sub set of other Value Set.
 - Ability to apply restrictions like term, coding scheme version, etc and resolve Value Set and Pick List Definition.
 - Ability to list all the coding schemes referenced by a Value Set Definition.

- Ability to list all Value Set Definition URIs that references a Coding Scheme. Etc.
- Authoring functions :
 - Ability to load/update/remove Value Set Definition, DefinitionEntry and Properties using Versioning API.
- Scripts :
 - LoadValueSetDefinition(.bat and .sh) –Loads Value Set Definition in LexGrid XML file into repository.
 - LoadPickListDefinition(.bat and .sh) – Loads Pick List Definition in LexGrid XML file into repository.
- GUI :
 - New GUI (LB_VSD_GUI in gui folder of LexEVS API install directory) is developed to test all the functionalities that are available for Value Set Definitions. This tool should mainly be used for testing purpose and not in production. Here are few major functions available through this GUI tool:
 - Create new Value Set Definition
 - Update existing Value Set Definition
 - Remove existing Value Set Definition
 - Load Value Set and Pick List Definition from LexGrid XML file
 - List all Value Set and Pick List Definitions present in the system
 - Add/update/remove DefinitionEntries (rule sets)
 - Add/update/remove properties
 - Resolve Value Set and Pick List Definition using specific set of coding scheme and version that are loaded in the system
 - Ability to validate if the changes made returns desired result before saving changes to the database
 - Export Value Set and Pick List Definitions in LexGrid XML format
 - Following query functions are also available:
 - Filter by Coding Scheme Reference
 - Filter by Concept Domain
 - Check if concept code is valid entry in Value Set Resolution
 - Filter by a term
 - Check if one Value Set is subset of other
 - Filter by Value Set Definition Name
- JUnit :
 - Junit to test all the Value Set functionalities are included in lbTest java package.
 - Org.LexGrid.valuedomain.test.VDAllTests.java – Loads test data, runs both Value Set and Pick list functionality tests, and removes loaded test data.

Note: Usage Context and Concept Domain is not included in this release, but will be in next release.

Versioning (gForge # 26651)

Versioning is a new functionality that has been added to LexEVS API in this release. Versioning API is divided into two parts, loader and query API. For this release, loader API is included; query API will be included in next release.

- Loader functions:
- Loader enables the editing capability of a codingScheme, valueset and picklist.
 - Changes can be applied in XML format.
 - Changes can be applied in the form of LexGrid Java castor objects as well.
 - Loader maintains the history of changes applied to the versionable objects.
 - Types of changes that can be applied on a versionable object are

NEW – to create a new versionable element

MODIFY – to change the attributes of an existing versionable element

VERSIONABLE – to change (or schedule a change of) the status of a versionable element within the context of the containing service.

REMOVE – to remove a versionable element from the service. (Note that the versionable object will be removed completely from the system including the history. VERSIONABLE Retire should be used if the element and its history should remain)

DEPENDENT – no changes are to be made to the named element itself, but a versionable element whose identity is dependent upon this element is to undergo a change.

- Loader validates each of the versionable object before loading it. Below are the criteria for a valid versionable object.
- EntryState must be present.
- If the change type is NEW
- The object being loaded should not exist in lexEVS system. (Exception : The <Versionable Object> being added already exist.)
- prevRevision of entryState should be null. (Exception : Changes of type NEW are not allowed to have previous revisions.)
- If the change type is other than NEW
- The object being revised should be present in the lexEVS system. (Exception : The codingScheme being revised doesn't exist.)
- The object must have prevRevision, except when the versionable object is revised first time after the "initial load". (Initial load : CodingScheme/ValueSet/PickList loaded with out a user defined revision)
- The prevRevision should match the newest revision id of the given versionable object that is already loaded in the lexEVS system. (Exception : Revision source is not in sync with the database revisions. Previous revision id does not match with the latest revision id of the <Versionable object>. Please update the authoring instance with all the revisions and regenerate the source.)
- JUnit :
 - VersionableEventAuthoringTest in lbTest package.

XML Loaders

- Streaming XML (gForge # 26652): Ability to stream vocabulary contents in LexGrid XML format into LexGrid repository.
- Ability to load contents at following entry points:
 - Coding Scheme
 - Value Set Definition
 - Pick List Definition

- Revision
- JUnit :
 - Junit to test above functionalities are included in lbTest java package.

XML Exporter

- Content can be exported to a LexGrid XML file by use of the LexGrid XML Exporter.
- The LexGrid XML Exporter can be accessed from a command line or from the LexGrid GUI.
- As of Prototype 3, the following content export scenarios are supported from the command line, GUI or both:
 - Entire code system (both)
 - Concepts only (GUI)
 - Associations only (GUI)
 - Specific Association (GUI)
- Accessing function within the GUI:
 - Entire code system:
 - At the menu: Commands -> Enable Admin Options
 - Select code system
 - At the menu: Export Terminology -> Export as LexGrid XML
 - Concepts only:
 - Select code system
 - Click 'Get Code Set' button
 - Select the CodedNodeSet (CS) from the list
 - Click the 'LgExport' button
 - Associations only:
 - Select code system
 - Click 'Get Code Graph' button
 - Select the CodedNodeGraph (CG) from the list. **Note:** restrict the set of associations by name:
- Click the 'Add' button in the Restrictions frame
- Then, in the 'Configure LexBIG' window:
- Select 'Restrict to Associations' in the 'Restriction Type' drop down
- Select the association name in the 'Associations' selection box
- Click the OK button
- Click the 'LgExport' button

Know issues/restrictions:

- Command line
 - Does not currently support the same level of function as the GUI. Only entire code systems can be exported via the command line. Additional filtering options will be available from the command line at a later date (Alpha 1).
 - When calling the XML Exporter from the command line always specify the force option '-f'. Otherwise the export will fail saying the output file already exists. This is a bug and will fixed during Alpha1 development.
- Scaling issues. Exporting of concepts should take place via streaming the content to the file and so should not be constrained by memory. At this time, however, associations are loaded into memory and so memory may be a factor in exporting code systems with many associations. This will be addressed during Alpha 1 development.

GForge Items

21720 – Load MRMAP data

- MRMAP data is now loaded as a Mapping Coding Scheme

21935 – OWL loader processing of <owl:Restriction>

29177 Value Set query enhancement

27844 ISO 21090 Data Type Support

- All Analytical Grid Service methods use ISO 21090 Data Types

23770 - OBI.owl loading – incorrect curation status

23806 - OBI.owl loading – incorrect Imported from

22036 - Focus code not found when referencing external coding scheme

28645 - Manifest loader/metadata loader

26637 - Page CodedNodeGraph Results:

- CodedNodeGraph results are retrieved on demand from the database. Depth and Breath of the graph are both expanded on demand, so graph traversal time should not be effected by ontology, number of AssociatedConcepts, resolve depth, etc.
- AssociatedConcepts (and Resolved Entities, if needed) are retrieved in batches, instead of individually.
- Limitations - Duplicate AssociatedConcepts may exist in the graph to prevent cycles. This is consistent with LexEVS 5.x.

28460 - Load source qualifiers for generic properties in NCI-META RRF loader

- This was implemented in the 5.1.x meta loader, and will be included in the 6.0 meta loader (which is in progress – not included in prototype 3)

23643 - Java 1.6

- LexEVS 6.0 is required to be compiled and run in a Java 6.0 environment.

27026 - Modify Meta Browser Extension to support browsing and searching of other.

- Meta Browser Extension now pages Associated Concepts (much like the NCI Term Browser Extension) and includes a JSON processor for outputting path-to-root and neighborhood Entities. Implemented initially for LexEVS 5.1.x

25681 - Problem loading MeSH and Spanish version of SNOMED

- All RRF loaders are adjusted to load in UTF8 format - allowing for Spanish/French/etc characters.

28167 - ClaML Loader update for LexEVS 6.0

- Converted the model framework from emf to castor.

26975 - PropertyLinks won't load without namespace being supplied

- Fixed by adding a validation function, which can fix the nullentity entitycodenamespace field issue

26976 - Loader should not shut down on a duplicate property

- Fixed by adding a validation function which can handle the duplicatedproperty issue by removing one of them

26977 - Loader should not shut down on missing propertylink reference

- Fixed by adding a validation function which removed the propertylinkif the property does not exist

26072 -Incorrect Hibernate mappings of Oracle CLOB database type in caCORE SDK functions

- Fixed by changing the clob type to text in the hibernate mappingfiles.

27021 – Rename Value Domain

- Renamed Value Domain to Value Set Definition and Value Set Resolution.

24037 – Include immediate parent/children when Value Set Definition has transitiveClosure as 'false' and referenceAssociation is present.

22296 - OWL loader does not load concepts with paranthesis

- This issue has been resolved in LexEVS 6.0 Prototype 3 and later releases. Tested and verified.
- JUnit Test case is created and successfully executed with implementation class "org.LexGrid.LexBIG.Impl.bugs.GForge22296" in project lbTest.

LexGrid model changes

Package ValueSets:

- Renamed element ValueDomainDefinition to ValueSetDefinition and container ValueDomains to ValueSets. GForge # 27021.
- Added attribute 'conceptDomain' of type 'conceptDomain' to class 'valueSetDefinition' to support CTS 2 SFM.
- Added class 'propertyReference' and 'propertyMatchValue' to support CTS 2 SFM.
 - propertyReference contains : 'codingScheme'(Required), 'propertyName', 'propertyMatchValue'
 - propertyMatchValue extends from 'text' and contains attribute 'matchAlgorithm'
- Made 'definitionEntry' versionable to support CTS 2 SFM.
- Made class 'valueSetDefinition' and 'pickListDefinition' as entry point. This change allows to load and export Value Set Definition and Pick List Definition individually.

Package Naming:

- Added 'supportedConceptDomain' to support CTS 2 SFM.

- Added attributes 'codingScheme', 'entityCodeNamespace' and 'entityCode' to 'SupportedAssociation'. This was added to make it easier to get an AssociationEntity instance for a given associationPredicate.
- Added attribute 'propertyType' to 'SupportedProperty'. This provides an easy way to find all supportedProperties information by propertyType which are 'presentation','definition','comment' and 'property'. NCI requirement Gforge # 24699.

Package CommonTypes:

- Added 'conceptDomain' in localIds. Since ConceptDomain has been added to the model, we will need to maintain all the ConceptDomain used in the system in our SupportedAttributes mappings.
- Changed 'owner' in 'versionable' from 'source' to 'tsCaseIgnoreIA5String'. This was changed since other attributes of Source were never used.

Package Concept:

- Removed elements 'Concept' and 'Instance'. These first class elements are not required anymore as an 'Entity' can represent either 'concept' or 'instance' or any other type by specifying it in its EntityType.

Package Relations:

- Made class 'relation' versionable to Support CTS 2 SFM.
- Added 'isMapping', 'sourceCodingScheme', 'sourceCodingSchemeVersion', 'targetCodingScheme', 'targetCodingSchemeVersion' to class 'relation' to support loading of MRMAP RRF file.
- Added 'properties' container to 'relation' container to support loading of MRMAP RRF file.
- Removed 'isNative' and 'source' from 'relation' as they were never used.
- Added new class 'associationPredicate' with required attribute 'associationName' to support loading of MRMAP RRF file.
- Renamed class 'association' to 'associationEntity'. Removed all Booleans except 'isTransitive', and 'isNavigable'. And made 'associationEntity' subclass of 'Entity'. This change was also made to support loading of MRMAP RRF file.

Package Versions:

- Changed 'changeAgent' in class 'revision' from 'source' to 'tsCaseIgnoreIA5String' as none of the 'Source' attributes were used.
- Added unique key constraint to releaseURI column of systemRelease table.
- Added missing entryStateGuid to propertyLinks table.