NCI Extensions to Protege

The information and links on this page are no longer being updated and are provided for reference purposes only.

NCI Extensions to Protégé

Contents of this Page

This page is meant as an explanation of the NCI Protégé Extensions, why they were implemented and what they do.

The Editor's Guide for the most recent version of the NCI Protégé extensions can be found on GForge. This provides the user with instructions on how to edit within the NCI Edit tab, how to do searches using the Advanced Search tab, how to handle workflow, pull reports, etc.

EVS Protégé Extensions Introduction and History

Version 1.2.3.25 is the first public release of the EVS Protégé Extensions. There have been multiple revisions released, tested and used internally over the past year. The current version is considered stable and reliable and suitable for external release.

These extensions were created in response to a need for an open-source editing tool that could be customized to the needs of the EVS content editors. In addition, EVS wanted the ability to exchange and share editable content with external collaborators. The Protégé editing tool, developed by Stanford, was used as the basis and extended to meet these needs.

The extensions are built on the Protégé 3.1 codebase and utilize pellet 1.5. Some improvements to the Protégé software were required to meet the needs of EVS, but these have already been merged into the Protégé trunk. Both Protégé and Pellet are included within the download. The Explanation Server was developed in collaboration with Clark & Parsia LLC.

The Protégé and the extensions are built and expect to run on Java 1.5.

EVS Protégé Extensions packages

The EVS Protégé Extensions are broken into three separate packages. The Explanation Server provides classification and explanation functionality. The Protégé Server provides access to a central editing project for multiple users. The Protégé Client is the end-user application for accessing the Protégé Server and editing content.

Explanation Server

The Explanation Server is used to do classification of the ontology and provide explanations on demand to the Protégé Client gui. It runs directly against the database, independently of the Protégé Server. The Protégé Server queries the Explanation Server for information when needed and controls requests for classification, so multiple clients cannot try and classify at the same time.

Protégé Server

The Protégé Server provides multi-user access to a single ontology stored in a database. It coordinates and controls user activities and resource allocation. It stores a history of user actions for use in tracking changes and resolving conflicts. The server also provides a centralized means of enforcing business rules and configurations upon client applications. EVS has extended the Protégé Server to allow tracking of workflow and assigning of editing tasks.

Protégé Client

The Protégé Client is a java-swing based gui used for editing an ontology. EVS uses the client to connect to the Protégé Server application, allowing multiple editors to share the same ontology. The Protégé client can also be used in standalone mode to edit a single ontology but some of the client-server specific extensions are then disabled. The client is used in standalone mode by managers to perform Prompt comparisons and terminology exports.

Protégé Business Rules

The following are EVS business rules that the NCI Edit tab was programmed to enforce. These rules are a major reason why it was necessary to write an extension rather than using the default Protégé editor.

Rules for Editing

- Each concept must have one and only one NCI/PT Full-Syn
- Each concept must have one and only one Preferred_Name
- HD, AQ and PT are equivalent Full-Syn term types. (i.e. an NCI/AQ can take the place of an NCI/PT)
- NCI/PT value must match the Preferred_Name
- Properties cannot be duplicated within a concept
- Roles cannot be duplicated within a concept
- Role groups cannot be duplicated within a concept
- Superclasses cannot be duplicated within a concept
- The last superclass of a concept cannot be deleted
- Cannot create a restriction pointed at a retired, pre-retired, or pre-merged class
- If concept has Def_Curator property present, only authorized users can create, modify or delete the DEFINITION
- If concept has Def_Curator property present, only authorized users can modify or delete the Def_Curator
- Def_Curators must exist in a configuration file. No "dummy" Def_Curators
- Each DEFINITION must have 1 review date, 1 review name, 0 or 1 attributes, and no source
- Only single spaces allowed in a DEFINITION unless preceeded by !, ? or .
- No leading or trailing spaces allowed in properties
- Only single spaces allowed between tokens in properties (except for DEFINITION)
- Properties cannot include the characters :,!,? or @ (except for DEFINITION which allows ! and ?)

Retire

- Pre-retired concepts are retired under the preretired_concepts bin
- Pre-retire process forces dependent classes to be fixed (children retreed, incoming roles re-pointed)
- Pre-retire concept must have and Editor_Note added explaining the retirement
- Only the lead editor may do the final retire on a concept
- Only the lead editor may edit retired concepts
- Roles in the retiring concept converted to annotation properties called OLD_ROLE
- Retiring concept deprecated using owl:deprecatedClass

Merge

- Pre-merge retiring concepts are double-treed under premerged_concepts bin
- Pre-merge identifying properties (Merge_From, Merge_To) are added to the surviving and retiring concepts
- Pre-merge retiring concept must have an Editor_Note added explaining the merge
- Only the lead editor may do the final merge on a concept
- Non-redundant roles and properties of the retiring concept are added to the surviving concept
- Non-redundant incoming roles pointed at the retiring concept are re-pointed at the surviving concept
- Roles in the retiring concept converted to annotation properties called OLD_ROLE
- Retiring concept deprecated using owl:deprecatedClass

History Record Generation

All edits done on the vocabulary are recorded in an audit log in the database. The data recorded includes the username, type of edit, date and time of edit, and reference concepts. This raw audit log is processed to remove any identifying information using Prompt. The scrubbed version is called concept_history and is published on a monthly basis.

The fields in these files are conceptcode|editaction|editdate|referencecode.

- conceptcode is the unique, permanent, alphanumeric identifier of the concept
- editaction is the activity being recorded (create, modify, split, merge, retire)
- editdate is the date the activity occurred
- referencecode is the identifier for a concept affected by or influencing the action, as detailed below

The record will normally just be a single line and will look like this:
C#####|create|dd-mon-yy|(null) or
C#####|modify|dd-mon-yy|(null)

Special Edits

In the particular cases of Split, Merge and Retirement there are multiple rows written with reference codes included.

Splits

In a split, a single concept is split into two. The original concept survives and a new concept is generated. Two split records are written for the original concept with reference codes for the resulting concepts and a create history record is written for the new concept. In the case of C11111 being split into C11111 and C22222 the history will appear as follows:
C22222|create|dd-mon-yy|(null)
C11111|split|dd-mon-yy|C22222
C11111|split|dd-mon-yy|C11111

Retires

In a retirement the concept is moved from its old location in the tree hierarchy into the Retired_Kind. A retire record is written for the retiring concept with a reference code of the old superconcept. If a concept has multiple superconcepts, then a retire record is written for each reference. In the case of retiring concept C11111 which has two superconcepts (C22222 and C33333), the history will appear as follows:
C11111|retire|dd-mon-yy|C22222
C11111|retire|dd-mon-yy|C33333

Merges

In a merge, two concepts are merged into one. One of the two concepts survives and the other concept is retired.
A merge history record is written for both of the concepts with a reference code of the surviving concept and a retire record is written for the concept that retires. In the case of C11111 merging with C22222 and C11111 surviving, the history will appear as follows:
C11111|merge|dd-mon-yy|C11111
C22222|merge|dd-mon-yy|C11111
C22222|retire|dd-mon-yy|(null)

Protégé Editing History Application

There are a couple of simple web applications that are made available to allow editors to review the evs_history and concept_history tables. The evs_history is written in real-time, as edits are occurring. The concept_history is written during the Prompt cycle. The applications allow access to both the Production and QA tiers of both BiomedGT and NCI Thesaurus.

- EVS History Query
- Concept History Query