Managing Ontology Life Cycle: Part I

Jennifer Vendetti
Stanford Medical Informatics
Stanford University

8th International Protégé Conference
Madrid, Spain, July 2005

Tasks in the life cycle of ontology and knowledge base development

- How to reuse or import existing resources?
- How to visualize information in the knowledge base?
- How to manage multiple ontologies using Prompt
- How to query or search knowledge bases?
- How to set up Protégé for multiple users
- How to export to external formats?
- How to add and test integrity constraints?
Example Scenario

- **Goal:** Develop a medical decision-support application that generates recommendations based on clinical practice guideline.
- **Guideline example:** Management of common cold.
- **Disclaimer:** Tutorial example, no real medicine involved.

Example Domain

- A **guideline** is a set of **recommendations** consisting of:
  - **contexts** (e.g., presentation of symptoms)
  - **tasks**
    - **actions** (e.g., inquiry, home care or referral)
    - **decisions:** choice of action based on **preference** criteria (e.g., symptoms of serious problem)

- **Patient state** encodes information about a particular patient (e.g., Observations, prescribed medications, etc.)

- **Concepts** represent abstractions of medical conditions (e.g., cough, fever, laryngitis).
Example ontology in Protégé

Tasks in the life cycle of ontology and knowledge base development

- How to reuse or import existing resources?
- How to visualize information in the knowledge base?
- How to manage multiple ontologies using Prompt
- How to query or search knowledge bases?
- How to set up Protégé for multiple users
- How to export to external formats?
- How to add and test integrity constraints?
Import & reuse of existing resources

- **Issue:** Get existing resources into Protégé
- **Solutions:** Multiple
- **Things to consider:**
  - What formats are my resources in?
  - Import some or all?
  - Represent or reference?

Example import scenarios

- Import resources already in Protégé-compatible formats (existing Protégé projects, OKBC, RDFS, OWL)
- Import a database
- Import arbitrary XML files
- Import concepts from external servers
Import Protégé-compatible resources

Protégé Ontologies Library
- Download existing Protégé projects and use Protégé’s inclusion mechanism
- Submissions welcome !!

OKBC Tab
- Import OKBC-compliant ontologies
- Downside: few exist, unsupported

Demo: DataGenie Tab

Import entire database
- Tables map to classes
- Columns map to slots

Downside: no export
Demo: XML Tab

- Top level elements map to classes
- Contained elements map to slots

Downside: Unsupported, undocumented

UMLS Tab

- National Library of Medicine’s Unified Medical Language System

- Biomedicine and health related concept databases (concepts, their names, their relationships)

- Set of software tools to access databases
1. Patient reports some combination of the following symptoms:
   - Sore throat
   - Rhinorrhea
   - Cough
   - Fever, usually of < 102° lasting < 72 hours
   - Laryngitis

2. Determine age

Assess symptoms

Inflammation of the larynx. This condition presents itself with dryness and soreness of the throat, difficulty in swallowing, cough, and hoarseness.
Other import plug-ins

- OntoBase – read, navigate, update arbitrary databases
- WordNet Tab – import lexical content from WordNet
- Scripting Tabs (JessTab, Algernon, Protégé Script Console)
  - Scripting for... well... anything really (covered by Samson)
- Apelon DTS Plug-in
  - Commercial plug-in to browse/reference terminologies from Apelon’s Distributed Terminology Server (SNOMED CT, LOINC, etc.)

Visualization of knowledge bases

- What do we mean by visualization?
- What are the issues in visualizing ontologies?
- Large-scale visualization
- Visualization of non-standard data types
- Customization of instance display
Visualization of knowledge bases

- Visualization: graphically display data to facilitate better understanding of its meaning (http://www.twocrows.com/glossary.htm)
- Principles of good design
  - Appropriateness for the information content
  - Increased number of displayed dimensions
  - Increased data density

From: http://www.edwardtufte.com/tufte/posters
Knowledge base visualization issues

- What are the “meanings” that should be highlighted
- What are the alternative graphical displays for the “meanings”
- What are the appropriate navigation paths?

More specific display issues

- Selection of information
- Visual metaphors
- Scalability
- Domain specificity
- Possibility of multiple views
- Degree of user control
Protégé’s default visualization

- Tabs provides large-scale views
- Slot widgets provide default views of Protégé data types
- Default views highlight is-a (class/superclass), binary (slots), and instance-of (class-instance) relationships

Larger scale visualization

- OntoViz Tab – visualize ontologies with GraphViz
- Jambalaya – visualize ontologies with SHriMP (Simple Hierarchical Multi-Perspective)
- TGViz – visualize ontologies with TouchGraph
Jambalaya

- Uses SHriMP (Simple Hierarchical Multi-Perspective)
- SHriMP is designed to help people browse complex information spaces
- Upside: very feature rich
- Downside: bigger learning curve than other tools
- Documentation/tutorials: http://www.thechiselgroup.org/jambalaya
Jambalaya

TGViz Tab

- Utilizes TouchGraph (renders networks as interactive graphs)

- TouchGraph uses “Spring Layout”

- PubMed uses TouchGraph to visualize graphs of related documents in medical libraries
Visualization of non-standard data types

- **Protégé’s basic data types**
  - integer, float, string, symbol, class, instance

- **Non-standard data types**
  - Date
  - Image
  - URL
  - Ordered list
  - …

---

**Image Widget**

![Image Widget](image.png)
URL Widget

Viral Upper Respiratory Infection (VURI) in Adults and Children

Download: Viral Upper Respiratory Infection (VURI) in Adults and Children (617 KB)

Released 05/10/04

Scope and Target Population:
Children, adolescents, and adults who are in generally good health and are not at risk.

Creation Date Widget

Date:
Sat Jul 16 16:11:42 PDT 2005

Resources:
Customization of instance display

- Custom slot widgets are primary method
- Classes and their instance forms are fixed within a project
- Can override form customizations in including projects

Instance & Knowledge Tree Tabs

Instance Tree
- view instances of classes as root nodes of trees
- trees contain directly and indirectly referenced frames

Knowledge Tree
- designate a top-level instance and navigate a tree of “contained” instances
Instance Tree Tab

Knowledge Tree Tab
Graph Widget

- Focus on one example of a custom slot widget - the graph widget
- Detailed tutorial on our Web site: http://protege.stanford.edu/doc/tutorial/graph_widget/

What is the Graph Widget?

- Allows visual editing of instances and relationships between instances
- Alternative to Protege’s “Forms” for entering instance data
When is the graph widget appropriate?

- When instances of a slot are connected as values of some slots (e.g., a linked list where one instance is links to another through a slot relation)

- When instances of a slot are related by instances of Directed-Binary-Relation class

- Speeds knowledge acquisition in ontologies with heavily interconnected concepts.

- Helps convey meaning and organization of acquired knowledge.

- Data that resembles process diagrams, flow charts, organizational charts.
Graph Widget

What are other custom slot widgets?

- ContainsWidget
  - Embeds forms for slots of type instance

- InstanceTableWidget

- InstanceListWidget
Managing multiple ontologies

What is Prompt?

- Compare versions of the same ontology (version management)
- Move frames between included and including project
- Merge two ontologies into one
- Extract a part of an ontology

Demo: Prompt
Managing Ontology Life Cycle: Part II

Samson Tu
Stanford Medical Informatics
Stanford University

8th International Protégé Conference
Madrid, Spain, July 2005

Tasks in the life cycle of ontology and knowledge base development

- How to reuse or import existing resources?
- How to visualize information in the knowledge base?
- How to manage multiple ontologies using Prompt?
- How to query or search knowledge bases?
- How to set up Protégé for multiple users?
- How to export to external formats?
- How to add and test integrity constraints?
4. How to query or search knowledge bases?

- Problem: you have specific information you are looking for in the knowledge base
- Considerations
  - What is the search space?
  - What identifying information do you have?
  - What do you want to do with search result?
  - What is the efficiency of search?

Scenario: Given the name or display name, find the frame in Protégé browsers

- Example: Search for a class whose name begins with “recom”
- Solution: Use Protégé “binocular” search
Scenario: Given a string, search its occurrences in the KB

- Example: Search occurrences of the string “symp”
- Solution: Use the StringSearch tab

Scenario: Search for instances of classes whose slot values satisfy simple constraints

- Example: Search instances of Task whose label contains “ref” or “home”
Scenario: Find instances of a class that satisfy constraints involving other instances
- Example: Find nodes A in a directed graph such that A follows a decision and A is not the value of the decision’s “alternatives” slot
- Solution: Use PAL query

scenario.png

Scenario: Programming to search in the KB (scripting languages)
- Problem: You need to perform search not covered in available GUI tools
- Solution 1: Use interface to scripting languages (Jess, Algernon, Python, etc.)

(deffunction findApplicableGuideline (?pid)
  (return (find-all-instances ((?g Guideline))
    (hasApplicableContext ?g ?pid))))

(deffunction hasApplicableContext (...))
Scenario: Programming to search in the KB (search API)

- “High-level” Java API for different classes of searches
- Search based on “context” (KB, class tree, instance tree, …) and “conditions” (constraints on slot values)

**Example: InstanceTree Search for Instances**

InstanceTree: all instances that are referenced directly or indirectly from a given instance

In instanceTree ”Mock guideline for managing cold”, search for instances that have browser text name “home” and whose “code” slot has value class HomeCare. This search can be done at only 1 level, or recursively.

See Protégé search API documentation

---

Scenario: Programming to search in the KB (Protégé API)

- “match” methods in Protégé Java API *KnowledgeBase* interface
- Optimized for search database backend
Tasks in the life cycle of ontology and knowledge base development

1. How to reuse or import existing resources?
2. How to visualize information in the knowledge base?
   1. How to customize display of instances?
   2. How to display non-standard data types?
   3. What options are there to display portions of knowledge bases?
   4. What options are available for navigation and browsing?
3. How to manage multiple ontologies using Prompt
4. How to query or search knowledge bases?
5. How to set up Protégé for multiple users
6. How to export to external formats?
7. How to add and test integrity constraints?

5. How to set up Protégé for multiple users

- Problem: You have a Protégé knowledge base that you want multiple users (human and programs) to access or edit it simultaneously
- Considerations:
  - installation constraints (availability of network, thin or thick clients)
  - plug-in requirements
  - communication among users
Protégé web browser: Thin clients using mostly standard plug-ins

- Requires installation servlet-capable web server (e.g. Apache Tomcat) on server side
- Changes saved only with database backend
- Possible to add annotations
- Possible to get screen shots of Protégé GUI associated with each instance
- Possible to download projects
- Changes not propagated to different client browsers
- Configuration through metaproject

Multi-user Protégé: Thick client using all available plug-ins

- Protégé installations on both server and client side
- Changes saved with database backend and in-memory backends
- Configuration through metaproject
- Firewalls an issue
- Included projects an issue

Scenario: Centralized server not an option

- Example: Individuals or groups work separately with no guaranteed access to central server
- Approach: Use PROMPT to compare and merge projects

Tasks in the life cycle of ontology and knowledge base development

1. How to reuse or import existing resources?
2. How to visualize information in the knowledge base?
3. How to manage multiple ontologies using Prompt
4. How to query or search knowledge bases?
5. How to set up Protégé for multiple users
6. How to export to external formats?
7. How to add and test integrity constraints?
6. How to export to external formats?

- Problem: Your users (and applications) require formats different from Protégé’s backend formats
- Consideration
  - Entire Protégé knowledge base or just selected portions
  - One-way export or two-way conversions
  - The relative representational power of external formats
    - possible use of annotations to represent information that would be lost otherwise

Scenario: Entire Protégé knowledge base in two-way conversions

- Example: Request to make Protégé an RDF editor
- Protégé backends
  - Alternatives
    - XML schema – saves a Protégé project using a fixed Protégé XML schema
    - XML ontology – saves a Protégé project by creating an XML schema based on the ontology in the project
    - RDF, OWL – save in formats used in semantic web formats
  - Not always possible
    - Often mismatch in knowledge models
  - Difficult to build and maintain
    - Requires updates as Protégé evolves
Export ontology or KB as HTML (In Protégé 3.1)

- Configurable HTML export format
  - Select classes, slots, facets to export
  - Select header, footer, and stylesheet

Scenario: Export to UML (Unified Modeling Language)

- UML: Dominant modeling standard in software engineering
- Solution: UML “backend”
  - Not a true backend: lose information (e.g., slot overrides)
  - Export in XMI 1.4 format
  - Readable in UML tool such as Poseidon
Scenario: Export to database

- Use case: Application programs use relational database management systems in multi-tier architecture
- No generic mapping of Protégé knowledge model to relational schema
  - e.g., Terminological classes akin to data
- Protégé stance
  - Protégé database backend optimized for Protégé user interface
  - Responsibility of local developers to create application-specific database export

Scenario: Export to XML

- Use case: Available tools for manipulating XML-formatted content
  - e.g., Use of XSLT to publish in alternative formats
- Solutions
  - OWL: external conversion tool
  - Specialized approaches
    - XML tab
    - Experimental XML file format
XML tab as exporter

- Export all classes and instances in separate XML files

Experimental XML file format

- Uses Protégé XML schema

  http://protege.stanford.edu/xml/schema/protege.xsd

- Will be default file format for frame-based projects
Tasks in the life cycle of ontology and knowledge base development

- How to reuse or import existing resources?
- How to visualize information in the knowledge base?
  - How to customize display of instances?
  - How to display non-standard data types?
  - What options are there to display portions of knowledge bases?
  - What options are available for navigation and browsing?
- How to manage multiple ontologies using Prompt
- How to query or search knowledge bases?
- How to set up Protégé for multiple users
- How to export to external formats?
- How to add and test integrity constraints?

7. How to add and test integrity constraints?

- Problem: You want to verify that the statements encoded in your ontology and knowledge base satisfy some properties
- Considerations
  - What is the knowledge model (logic) of your ontology/KB
    - OWL: Java tests in Protégé OWL
    - Frame: slot constraints and PAL constraints
Validating integrity constraints

- Facet Constraint Tab
  - Protégé facets are constraints on values of slots (e.g. minimum cardinality)
  - FacetConstraint Tab brings all instances with facet constraint violations together in one place
- PAL Constraint Tab
  - Protégé Axiom Language (PAL) lets you write integrity constraints across multiple slots and multiple instances
  - PAL constraint tab allows checking of PAL constraints
- EZPAL Tab
  - Provides templates for easier authoring of PAL constraints

Violations of facet constraint are shown as slot widgets with red border
Facet-constraint tab brings together instances with facet-constraint violations

- Example: open any project
- Select Facet Constraints tab
PAL constraints

- Overview
  - What PAL is and what it can be used for
  - How PAL is integrated into the Protégé framework
- ezPAL: fill-in-template method to write PAL constraints
- Mechanics: How to write PAL constraints

Example

- All decisions are part of recommendations
  
  \[
  \text{(defrange \text{?decision} :FRAME Decision)} \\
  \text{(defrange \text{?recomm} :FRAME Recommendation)} \\
  \text{(forall \text{?decision}} \\
  \text{ (exists \text{?recomm}} \\
  \text{(recommendationParts \text{?recomm} \text{?decision}}))}
  \]
PAL: What’s It?

- What it is:
  - A constraint language that helps to enforce the semantic properties of knowledge bases encoded in Protégé
  - A query language for searching instances that satisfy certain relationships

- What it is not:
  - A general predicate-logic language
  - A way to do write rules in Protégé
  - Another way to write definitions of concepts modeled in Protégé

A limited first-order logic extension of Protégé

- We decided on a variant of Knowledge Interchange Format (KIF)
- We use the KIF connectives and the KIF syntax
  - =, /=, not, and, or, =>, forall, exists
- Not all the KIF constants and predicates are included
  - (defrelation ...), (deffunction...) are omitted
  - added Protégé-specific predicates

(forall ?decision
(exists ?recomm
(recommendationParts ?recomm ?decision)))
Difference between constraints and axioms

- Use the syntax of logic but have different semantics
  - Axioms are necessarily true
  - Constraints may be violated
- (exists ?y (mother-of John ?y))
  - Asserted as an axiom: John has a mother, even though there is no explicit object in KB
  - Asserted as a constraint: constraint is violated if no existing instance ?y satisfies the relation
- PAL: Protégé Constraint Language

PAL constraints

- Overview
  - What PAL is and what it can be used for
  - How PAL is integrated into the Protégé framework
- ezPAL: fill-in-template method to write PAL constraints
- Mechanics: How to write PAL constraints
Compatibility with Protégé framework (1)

- PAL constraints are themselves frames

Specialized editor for viewing and editing PAL constraints
Compatibility with Protégé framework (2)

- Quantified variables are declared as holding instances of a class
  - (defrange ?decision :FRAME Decision)
  - (forall ?decision …)
  - (exists ?decision …)
- Slots are predicates
  - (recommendationPart ?recom ?decision)
- Cardinality-single slots are functions
  - (label ?action) returns a string
- Additional relations (e.g., >, subclass_of) and functions (e.g., +, -, *, coerce-to-string) have been added

Enforcement of constraints is not real-time

- It’s not always possible for the user to always have a consistent KB while editing
  - And, even if it were possible, it might be inconvenient.
- Therefore, the user should decide when to check constraints
PALConstraints Tab: Allows specification and checking of complex integrity constraints

Bring available PAL constraints into view

Operations on selected PAL constraint

Select or unselect all PAL constraints

Click to evaluate selected PAL constraints

Select or unselect individual PAL constraints
Evaluate selected constraints

Instances with PAL constraint violations

Example of PAL constraint

PAL Query Language

- Taking a constraint and finding instances that violate it is much like finding instances that satisfy a statement
  - (not (exists ?x (...))
- Query engine introduces two keywords
  - find
  - findall

(findall ?node
  (exists ?decision (and
    (exists ?preference (and (:FROM ?preference ?decision)
      (:TO ?preference ?node)))
    (not (alternatives ?decision ?node))))
PAL constraints

- Overview
  - What PAL is and what it can be used for
  - How PAL is integrated into the Protégé framework
- ezPAL: fill-in-template method to write PAL constraints
- Mechanics: How to write PAL constraints

EZPAL tab

- Templates for fill-in-the-blanks method of defining PAL constraints
Exercise: Use of ezPAL

- Make sure that template.pprj is included
- Select property and mode
- Want to write constraint
  - Every instances of class Guideline must have a unique value for the `title` slot
- Select template, and fill in details

You should have

Click Translate to generate PAL
PAL constraints

- Overview
  - ezPAL: fill-in-template method to write PAL constraints
- Mechanics: How to write PAL constraints

How to write PAL constraints

- Preparation
  - Learn the syntax and available predicates and functions
  - Understand ontology
  - Write down constraints in natural language
- Declare variables
- Write constraint
- Check syntax
- Try out examples
- Iterate
Learn the syntax and available predicates and functions

- Best to re-write examples
- Use PAL editor
  - shows available predicates for selection
  - syntax checking

Understand ontology
Write constraint in natural language

- A recommendation must have at least one context and one action

Declare Variables

- Quantified variables refer to frames
- Syntax for explicit declaration
  - local variable (defrange ?action :FRAME Action)
  - global variable %action: Don’t use

(defrange ?recomm :FRAME Recommendation)
(defrange ?context :FRAME Context)
(defrange ?act :FRAME Action)

Protégé allows the use of free variable (undeclared variable) in a PAL constraint if the constraint is attached to a class. However this practice has unintended consequences. Always declare variables.
Write Constraint

(defrange ?recomm :FRAME Recommendation)
(defrange ?context :FRAME Context)
(defrange ?act :FRAME Action)
(forall ?recomm
    (and (exists ?context
          (recommendationParts ?recomm ?context))
         (exists ?act
          (recommendationParts ?recomm ?act)))
)

Debug PAL Constraints

- Use tools that help with balancing of parenthesis and with indentations
  - PAL editor does these and more!
- Try small examples
- Test components
- Trace built-in predicates and functions
- Write existence clauses as queries
Summary: PAL

- Integrity constraints on knowledge bases help you to catch errors
- Frame-based Protégé allows use of two types of constraints
  - Facet constraints: constraints on properties of a slot
  - PAL constraints: constraints on instances and their relationships
- Tools like EZPal tab, facet-constraint tab, and PAL-constraint tab help you to write and check constraints

Comprehensive documentation is available http://protege.stanford.edu/plugins/paltabs/pal-documentation/
Summary

- Protégé plugin architecture allows functionalities to be added to core Protégé
- Tasks in the life cycle of ontology and knowledge base development
  - How to reuse or import existing resources?
  - How to visualize information in the knowledge base?
  - How to manage multiple ontologies using Prompt
  - How to query or search knowledge bases?
  - How to set up Protégé for multiple users
  - How to export to external formats?
  - How to add and test integrity constraints?