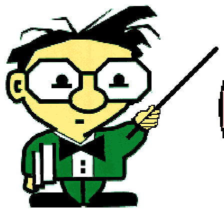
Achieving interoperability between WebODE and Protégé2000



Oscar Corcho

ocorcho@fi.upm.es

Ontology Group
Laboratorio de Inteligencia Artificial
Facultad de Informática. Universidad Politécnica de Madrid
Campus de Montegancedo sn,
28660 Boadilla del Monte, Madrid, Spain

Table of contents

The ontology translation problem

"The ontology translation problem appears when we decide to reuse an ontology (or part of an ontology) using a tool or language that is different from those in which the ontology is available"

- Translation problems
  - Different formats
    - Each language/tools stores and reads ontologies using different syntax
  - Different underlying KR paradigms
    - Frames, first order logic, description logic, semantic networks, conceptual graphs, and combinations
  - The knowledge lost in transformations should be preserved
    - In case that ontologies are restructured and translated back to the original language/tool

Current situation

- Existing approaches to ontology translation
  - Ontolingua
    - Ad-hoc translators, difficult to understand and maintain
      - Not documented
      - Translation decisions “hidden” in their code
  - “The family of languages” [Euzenat et al., 2002]
    - Formal approach, based on description logics
    - It does not deal with different formats
  - OntoMorph [Chalupsky, 2000]
    - Lisp-based tool to create translators more easily

- Why do we need better approaches?
  - Languages/tools evolve and new ones are created
    - RDF, RDF Schema, OIL, DAML+OIL, OWL
  - Translation decisions may change

Our proposal

- What do we need?
  - Declarative specifications of translations
    - Translators are automatically created from them
  - Specifications of knowledge transformations at different levels
    - Knowledge level
    - Symbol level
      - Syntax level
      - Lexical level
    - User interface
      - Semiotic level

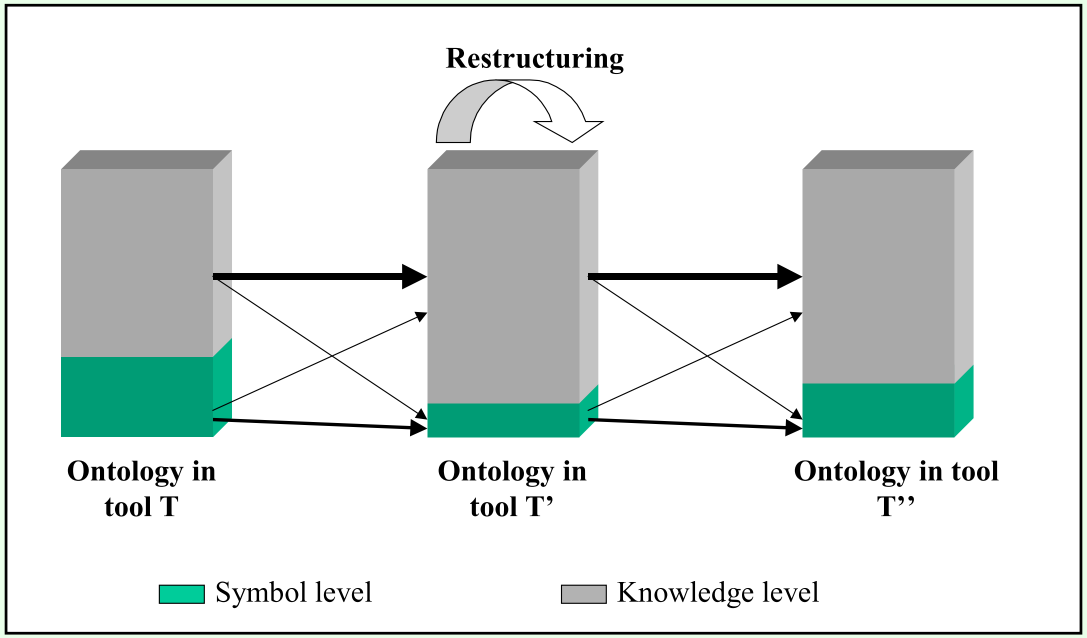


Table of contents

What is WebODE?

- Ontology engineering workbench
  - Scalable and extensible
    - Application server and database storage
    - Plugable Services
  - Multiple users support
  - Technology support for METHONTOLOGY
    - Most of the activities of the ontology lifecycle
    - Conceptualization, documentation, reasoning, import/export, evaluation, merge, etc.
  - Ontology interoperability
    - Java API
    - Translation services
  - Ontology-based applications
    - ODE-KM (Knowledge Management)
    - ODESeW (Semantic Web portals)

- WebODE homepage: http://webode.fi.upm.es/

WebODE. Knowledge model

Arpírez, J.C.; Corcho, O.; Fernández-López, M.; Gómez-Pérez, A. *WebODE: a Scalable Ontology Engineering Workbench*. **First International Conference on Knowledge Capture (KCAP-2001).** Victoria, Canada. 2001.

- Concepts
  - Class attributes
  - Instance attributes
- Concept groups (disjoint concepts)
- Relations
  - Built-in relations
    - Taxonomy: subclass-of, disjoint-decomposition, exhaustive-decomposition, partition
    - Mereology: transitive-part-of, intransitive-part-of
  - Ad-hoc binary relations
  - Relation properties (symmetry, transitiveness, etc.)
- Formal axioms and rules
- Instance sets (instances of concepts and relations)
- Constants, imported terms (URL based), bibliographic references, etc.

Protégé knowledge model

- Classes and metaclasses
  - Attached template slots (which can be inherited)
  - Own slots (obtained from metaclasses)
- Slots (first-class components)
  - Facets (predefined and ad-hoc)
- Taxonomy relations
  - Subclass of
  - Subslot of
- Class instances (individuals)
- PAL constraints and queries

Noy NF, Fergerson RW, Musen MA (2000). *The knowledge model of Protege-2000: Combining interoperability and flexibility*. **In: Dieng R, Corby O (eds) 12th International Conference in Knowledge Engineering and Knowledge Management (EKAW'00). Juan-Les-Pins, France. Springer-Verlag (LNAI 1937)**

Some transformation proposals

- Create the KR ontology of WebODE in Protégé2000
  - Easy transformation process
  - No information lost in the process
  - The transformed ontology is not legible, usable by Protégé2000 users
- Transform the WebODE ontology to the built-in Protégé2000 knowledge model
  - Some knowledge lost in the transformation (bibliographic references, synonyms, acronyms, etc.)
- Transform the WebODE ontology to the built-in Protégé2000 knowledge model, and preserve lost knowledge
  - Knowledge is preserved in the transformation
  - The transformed ontology is legible and usable

Transformations at the knowledge level (I)

WebODE Concepts (I)

- Previous transformations
  - Hidden classes in Protégé:
    - :WebODESynonym
    - :WebODEAbbreviation
    - :WebODEReference
  - Hidden metaclass in Protégé:
    - :WebODEConcept (subclass of :STANDARD-CLASS)
      - Create template slot **:Synonyms** and attach it to :WebODEConcept.
      - Create template slot **:Abbreviations** and attach it to :WebODEConcept.
      - Create template slot **:References** and attach it to :WebODEConcept.

Transformations at the knowledge level (II)

WebODE Concepts (II)

For each WebODE concept

WebODE	Protégé-2000
Concept	*Class*
Name (max. 200 characters)	:NAME
Description (max. 2000 characters)	:DOCUMENTATION
Synonym	Instance of class :WebODESynonym (hidden class)
Name	:NAME
Description	:DOCUMENTATION
Abbreviation	Instance of class :WebODEAbbreviation (hidden class)
Name	:NAME
Description	:DOCUMENTATION
Axiom	:SLOT-CONSTRAINTS
Reference	Instance of class :WebODEReference (hidden class)
Name	:NAME
Description	:DOCUMENTATION
Instance attribute	*See table 3*
Class attribute	*See table 4*

Transformations at the knowledge level (III)

WebODE Concept groups (I)

- Previous transformations
  - Class in Protégé:
    - :PAL-DISJOINT-CONSTRAINT
      - **:DOCUMENTATION**: “Class that represents WebODE concept groups”
      - **:ROLE**: Concrete
      - **:DIRECT-TYPE**: :STANDARD-CLASS
      - **:DIRECT-SUPERCLASSES**: :PAL-CONSTRAINT
      - Create template slot **:groupConcepts** and attach it to the class.
        - » **Documentation**: “concepts in the group”
        - » **Type**: Class
        - » **Allowed parents**: :THING (that is, any class in the ontology)
        - » **Cardinality**: (2,N)

Transformations at the knowledge level (IV)

WebODE Concept groups (II)

For each WebODE concept group, create an instance of :PAL-DISJOINT-CONSTRAINT

- **:PAL-NAME**: the concept group name in WebODE
- **:PAL-DESCRIPTION**: the description of the concept group in WebODE.
- **:PAL-STATEMENT**: the following expression

```
(forall ?W (forall ?X (forall ?Y (forall ?Z
    (=> (and (:groupConcepts ?W ?X) (:groupConcepts ?W ?Y)
        (subclass-of ?Z ?X) (/= ?X ?Y))
            (not (subclass-of ?Z ?Y)))))))
```

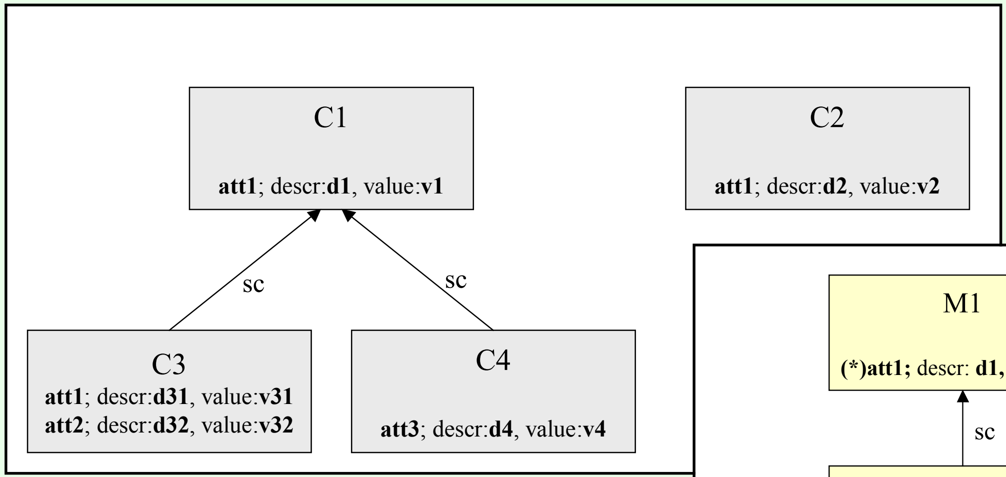
- **:PAL-RANGE**: the following expression

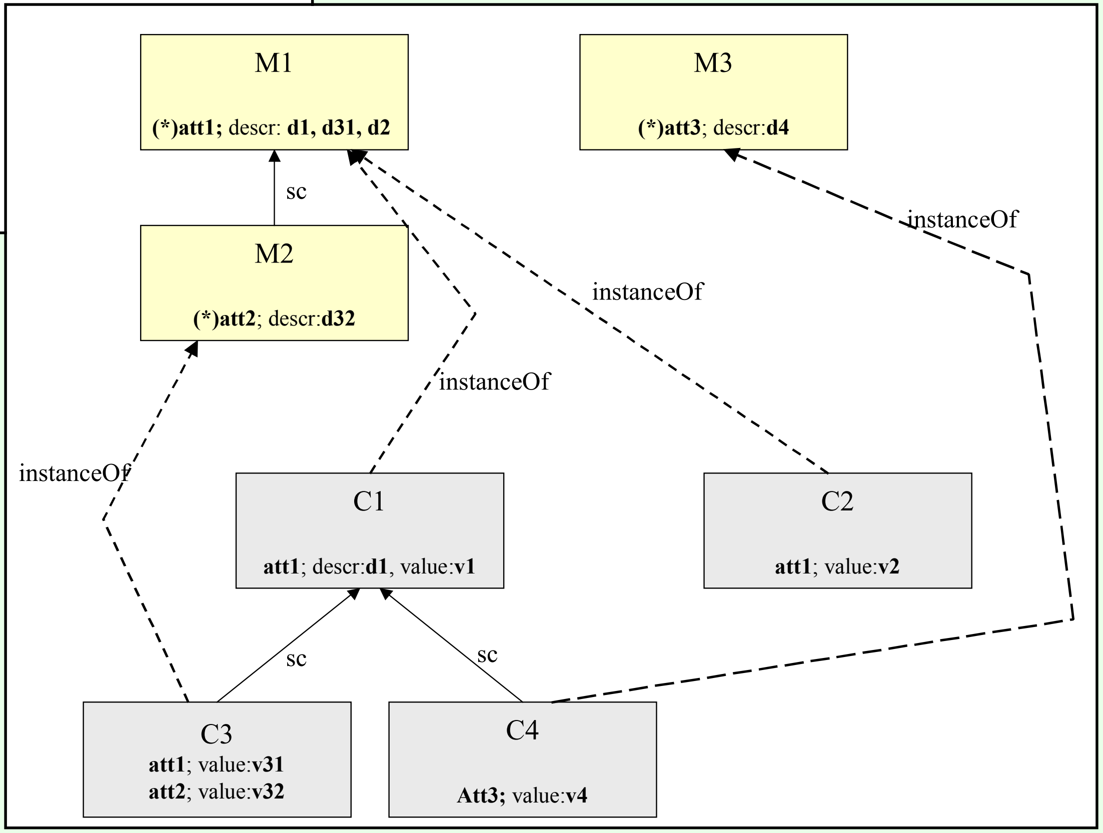
```
(defrange ?X :FRAME :STANDARD-CLASS)
(defrange ?Y :FRAME :STANDARD-CLASS)
(defrange ?Z :FRAME :STANDARD-CLASS)
(defrange ?W :FRAME PAL_DISJOINT_CONSTRAINT)
```

- **:groupConcepts**: the concepts in the concept group in WebODE

Transformation at the knowledge level (V)

- WebODE class and instance attributes





Transformations at the symbol level (I)

- Syntax transformations (slot value types)

WebODE	Protégé-2000
String	String
Integer	Integer
Cardinal	Integer. :SLOT-MINIMUM-VALUE set to 0, unless the minimum value of the slot is set explicitly to another different value
Float	Float
Boolean	Boolean
Date	String with SlotWidgetDate in form
Numeric Range	Float. :SLOT-MINIMUM-VALUE and :SLOT_MAXIMUM_VALUE have been already set
URL	String with SlotWidgetURL in form
Instance of Concept	Instance
--	Class
--	Any
--	Symbol

Transformations at the symbol level (II)

- Lexical transformations
  - Identifiers
    - WebODE allows using the same name for a concept and an instance.
    - Protégé automatically renames one of it
  - Maximum length for fields
  - Characters that are not allowed
  - etc.

Transformations at the user interface

- Customization of Protégé2000 forms
  - Removing WebODE-related slot widgets
  - Specific slot widgets for dates and URLs
  - Hiding classes from project
  - Activate PAL Constraint tab

- Import of a project
  - WebODEAdditions.pprj
  - It contains the WebODE additional components identified in "previous transformations"

Table of contents

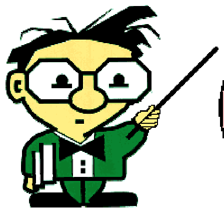
Future work

- Transformations from Protégé2000 to WebODE
  - Get back the knowledge preserved in transformation WebODE-Protégé
  - Maintain knowledge that could be lost in the transformation

- Declarative specifications of translators
  - Step 2:
    - Create a formal language from current tables
    - More automatization in translator creation

  - Step 3:
    - Ontology mapping between knowledge models
      - Based on existing approaches for information integration
      - Procedural attachments for “executing” mappings

Achieving interoperability between WebODE and Protégé2000



Oscar Corcho
ocorcho@fi.upm.es

Ontology Group
Laboratorio de Inteligencia Artificial
Facultad de Informática. Universidad Politécnica de Madrid
Campus de Montegancedo sn,
28660 Boadilla del Monte, Madrid, Spain