EROS: Explorer for RDFS-based Ontologies

Richard Vdovjak
Eindhoven University of Technology
PO Box 513, Eindhoven,
The Netherlands
richardv@win.tue.nl

Peter Barna
Eindhoven University of Technology
PO Box 513, Eindhoven,
The Netherlands
pbarna@win.tue.nl

Geert-Jan Houben
Eindhoven University of Technology
PO Box 513, Eindhoven,
The Netherlands
houben@win.tue.nl

Categories and Subject Descriptors
H.5.2 [User Interfaces]: Graphical user interfaces (GUI).

General Terms
Design, Experimentation, Human Factors.

Keywords
EROS, RDFS, GUI, Semantic Web.

1. THE CHALLENGE IN VISUALIZING RDFS-BASED ONTOLOGIES
RDFS is an acknowledged backbone of the Semantic Web architecture. Browsing and querying RDFS-based ontologies is becoming a characteristic (user) activity in Semantic Web applications. We argue that due to peculiarities of the RDFS language a new interface is needed to convey RDFS-based ontologies to the end-user in a comprehensible form.

The two main approaches currently used, the tree-based approach and the graph-based approach, do not fulfill the above requirement completely. The tree paradigm does not help the user in grasping other concept relationships than that used to construct the tree structure (most of the time being the rdfs:subClassOf relationship). The graph approach, on the other hand, displays all concept relationships but as a result introduces the full complexity of a directed labeled graph in which it is very difficult to spot the hierarchical structure of the ontology “hidden” behind the special kind of edges.

2. THE EROS APPROACH
EROS combines the tree-based and the graph-based approach into one, preserving the good aspects of both: the simplicity of the first and at least a part of the expressiveness of the second. The main idea behind EROS is to consider properties as partial mappings that relate (some) elements (classes) from the class hierarchy to either other (possibly identical) elements within the same hierarchy, or to a special element called Literal. Hence, the set of all elements from the hierarchy serves two purposes: first as a (potential) domain of all properties, second as their (potential) range.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ACM '03, January 12–15, 2003, Miami, Florida, USA.
Copyright 2003 ACM 1-58113-586-6/03/0001...$5.00.

Figure 1. The EROS UI, class-centric approach.

This inspired us to have two (almost) identical hierarchy trees in our interface, the left tree being the domain (“from”) tree, and the right tree being the range (“to”) tree extended with the Literal element. Properties themselves are depicted as arrows connecting the classes from the domain tree with the classes from the range tree. Cases of multiple inheritance are handled by displaying a list of super classes for the selected node. Note that though specifically designed for RDFS, this approach can be applied to any object/frame-oriented description language.

The EROS interface (Figure 1) is fully configurable in the sense that it allows to base the tree hierarchies not only on the rdfs:subClassOf relationship, but also on the rdfs:subPropertyOf relationship or any other transitive relationship chosen by the user. In this way, EROS can offer several views over the same ontology depending on the user preferences.

EROS serves both as a tool for exploring/browsing RDFS ontologies and expressing queries over those ontologies. The query creation process guides the user while composing an RQL query. This process consists of filling in the Select, Where and From clauses of a query in a point-and-click manner. For the actual query facilities we rely on an external RQL query engine.

EROS was implemented in Java and can be deployed either as a stand-alone application, or with some restrictions, as an applet. To verify the aforementioned ideas, an ontology-based photo library application was used as a test case.