Bringing the Semantic Web closer to the User

Richard Vdovjak

Peter Barna

Geert-Jan Houben

Flavius Frasincar

Technische Universiteit Eindhoven
P.O.Box 513, NL-5600 MB, Eindhoven, the Netherlands
{richardv,pbarna,houben,flaviusf}@win.tue.nl

Categories and Subject Descriptors

H.5.2 [**User Interfaces**]: Graphical user interfaces(GUI)

General Terms

Design, Experimentation, Human Factors

Keywords

RDFS, Semantic Web, RQL, User Interface, EROS

1. INTRODUCTION

As the current Semantic Web (SW) activities focus mainly on making the Web machine understandable, the end-user needs are often neglected. Providing an appropriate entry point for the end-user to the technologies introduced by the SW activity would be beneficial both for the end-users and the SW developers. Taking advantage of the semantic descriptions available on the SW end-users would achieve for instance much better search results and the SW developers would get a real-life feedback which is necessary for shaping and improving SW applications.

To bring the advantages of the SW to the end-user, one clearly needs an interface which would allow the user to explore, browse, and query the content he is interested in. RDFS [1] is an acknowledged backbone of the SW architecture. We argue that due to its peculiarities a new interface metaphor is needed to convey RDFS-based ontologies to the end-user in a comprehensible form. In this work we present such a metaphor and its implementation called EROS[1].

2. VISUALIZING RDFS

The problem of visualizing RDFS lies in the fact that it is difficult to show the whole expressive power of RDFS and at the same time to keep the user interface (UI) still comprehensible, easy to use, browse and navigate.

The two main approaches currently used, the tree-based approach and the graph-based approach (e.g. used in [4] and [3], respectively), do not in our opinion address the above issues completely. The tree metaphor, though very familiar as UI, does not help the user in grasping other concept relationships than that used to construct the tree structure (most of the time being the *subClassOf* relationship). The graph metaphor, on the other hand, displays all concept relationships but as a result introduces the full complexity of a directed labeled graph in which is very difficult to spot the hierarchical structure of the ontology "hidden" behind the special edges and not reflected by the position of the class nodes.

Combining the advantages of the two mentioned UI metaphors, one would desire the simplicity of a browsable tree and at least a part of the expressiveness of the graph based approach. This is exactly what we tried to achieve by the EROS interface.

3. THE EROS INTERFACE

The main idea behind this interface is to consider RDF properties as partial mappings that relate some elements (classes) from the class hierarchy into other (possibly identical) elements within the same hierarchy. Note that the set of all elements from the hierarchy serves two purposes: firstly as a (potential) domain of all properties and later as their (potential) range. This double purpose inspired us to have two identical hierarchy trees in our interface, the left tree being the domain ("from") tree, and the right tree being the range ("to") tree. Properties themselves are depicted as arrows connecting the classes from the left / domain tree with the classes from the right / range tree. In this approach one can display for a certain property at the same time both the context (the neighboring class hierarchy) of the domain class and the context of the range class. Cases of multiple inheritance are handled by displaying a list of super classes for the currently selected class. This approach, illustrated in Figure 1(left) can be considered as "class-centric" since the key transitive property on which both trees are built is the *subClassOf* relation.

Properties in RDFS are first-order citizens; if the user prefers to view the ontology with the "property-centric optics" and desires to explore the tree hierarchy based on the relation *subPropertyOf* the EROS interface can easily accommodate this demand by imposing that the left tree hierarchy is built based on the *subPropertyOf* relation, while the right tree (still hierarchically based on *subClassOf*) represents the domains and ranges of the properties from the left tree as depicted in Figure 1(right).

In fact, EROS can go even further by allowing the user to adapt the interface to his needs by choosing an arbitrary transitive[2] property as a key for building the left tree and align it with the right (*subClassOf*) tree. One can, for instance, choose to base the left tree hierarchy on the *wordnet:hypernym* relation and study the (possibly subtle) differences between the two tree hierarchies and also their mutual relations in terms of properties which connect resources from one tree to another. This feature comes at hand especially

[1]Explorer for RDFS-based OntologieS

[2]Transitive from the user's point of view.

WWW2003, May 20–24, 2003, Budapest, Hungary.
ACM xxx.

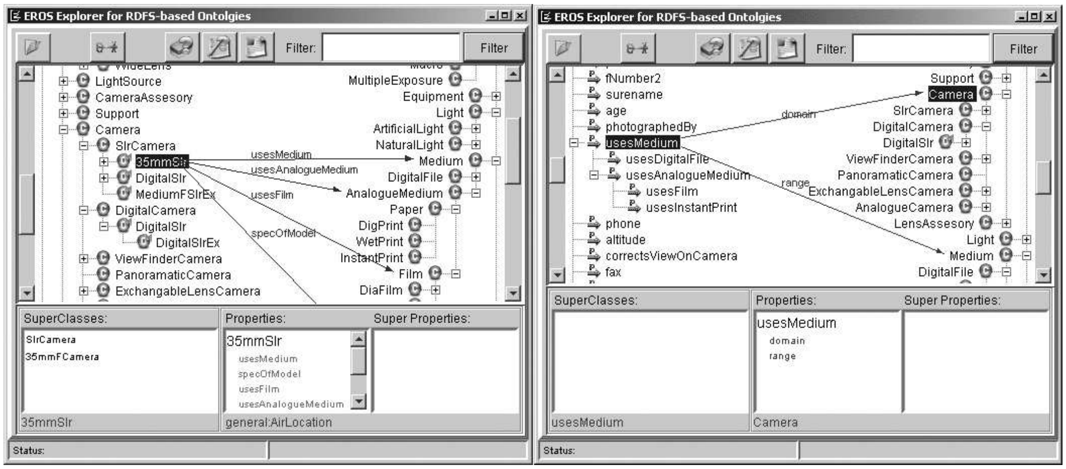

Figure 1: The EROS interface: Class-centric view, Property-centric view

in issues like integrating or aligning ontologies, which is a complicated process and EROS can (partially) offer support for this.

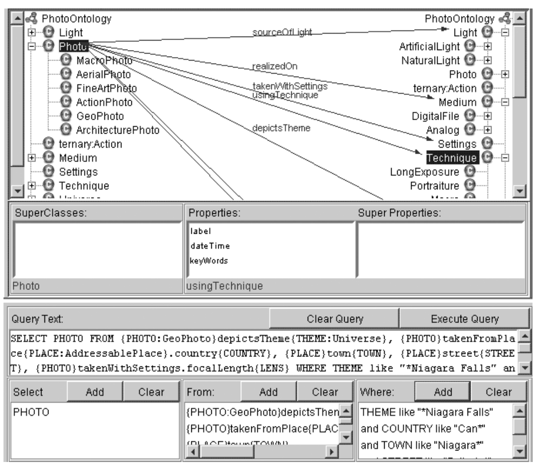

Figure 2: The EROS interface: Query Building Mode

4. QUERY GENERATION SUPPORT

When in query mode, the EROS interface, guides the user in building RQL [2] queries, i.e. specifying the `SELECT-FROM-WHERE` clauses. A new query formulation starts with a `FROM` part where the EROS interface assists the user in the formulation of a single path expression of the form: $\{VAR_i : DC_i\}p_i\{VAR_j : RC_j \mid L\}, \ldots$ This expression can be combined in a so-called chained expression which takes the following form: $\{VAR_i : DC_i\}p_i\{VAR_j : RC_j \mid L\}.p_j\{VAR_m : RC_m \mid L\} \ldots$, where $VAR_x : Y$ denotes a variable of type Y, p_x represents a property, and DC_i, RC_j denote classes serving as a domain and range respectively; L represents a literal.

EROS allows the user to generate such expressions by selecting a node in the graph (a variable of a certain type), then selecting a property of the chosen node that navigates the user to the destination node (again a variable). If the destination node is not of a literal type, the user can choose another property of that node traversing the graph further (building a chained path expression) or he can create a new path expression concatenating it with the previous ones.

Filling in the `SELECT` clause consists of choosing variables that are of interest from the list of variables introduced in the `FROM` clause. Similarly to the `SELECT` clause, the `WHERE` clause starts by selecting variables bound in the `FROM` clause and then building a Boolean expression by utilizing an offered list of appropriate operators. The queries that can be built in this way represent a large class of practical queries for ontology exploration and data retrieval. Figure 2 depicts the query building mode in EROS.

5. CONCLUSIONS

Essential in the use of the Semantic Web are the ontologies, often expressed in RDFS. They help to organize the resources in terms of their semantics, and thus offer a nice specification of the semantics of the entire application. However, for many applications this specification in terms of ontologies needs to be used by humans. The end-users employ such ontologies to search for terms and to mentally reason about these terms. Before they are constructing actual queries on the RDF metadata they need to familiarize with the ontology. For this purpose an effective visual representation of ontologies is vital. We addressed this issue providing a portable and light-weight[3] visual interface in which the user is able to view the ontology both from the viewpoint of classes and that of properties. This interface also assists in the generation of RQL queries.

6. REFERENCES

[1] D. Brickley and R. Guha. Rdf vocabulary description language 1.0: Rdf schema. W3C Working Draft 30 April 2002.

[2] G. Karvounarakis, V. Christophides, D. Dimitris Plexousakis, and S. Alexaki. Querying rdf descriptions for community web portals. In *17iemes Journees Bases de Donnees Avancees*, pages 133–144, 2001.

[3] A. Maedche, B. Motik, L. Stojanovic, R. Studer, and R. Volz. Ontologies for enterprise knowledge management. *IEEE Intelligent Systems*, January/February 2003.

[4] N. F. Noy, S. Michael, S. Decker, M. Crubezy, R. W. Fergerson, and M. A. Musen. Creating semantic web contents with protege-2000. *IEEE Intelligent Systems*, 16(2):60–71, March/April 2001.

[3] EROS is a Java Web application which downloads on the client's machine; the size of the whole package is approximately 350 Kb.