# OntoXpl[*] Exploration of OWL Ontologies

**Volker Haarslev and Ying Lu and Nematollah Shiri**
Computer Science Department
Concordia University, Montreal, Canada
haarslev@cs.concordia.ca
ying_lu@cs.concordia.ca
shiri@cs.concordia.ca

**Abstract**

This paper describes the OWL ontology explorer OntoXpl. It is available as a web server based on the tomcat architecture. Standard HTML browsers can be used to interact with OntoXpl. At least three potential user groups are targeted by OntoXpl's design: (i) users with a limited background of ontologies and OWL; (ii) ontology developers that are OWL experts; (iii) users interested in understanding and reusing existing ontologies. OntoXpl is intended to complement existing ontology editors and does not offer any editing support. The current implementation of OntoXpl is based on the OWL DL reasoner Racer and uses Racer's extensive query interface in order to support the exploration of OWL ontologies.

## 1 Introduction

Practical description logic systems play an ever-growing role for knowledge representation and reasoning research. In particular, the semantic web initiative [3] is based on description logics (DLs) and defines important challenges for current system implementations. Recently, one of the main standards for the semantic web has been proposed: the Web Ontology Language (OWL) [17]. OWL is based on two other standards: Resource Description Format (RDF [10]) and its corresponding "vocabulary language" RDF Schema (RDFS) [4]. In recent research efforts, these languages are mainly considered as ontology representation languages (see e.g. [1] for an overview). The languages are used for defining classes of so-called abstract objects. Now, many applications start to use the RDF part of OWL for representing information about specific abstract objects of a certain domain. Graphical editors such as OilEd [2] or Protégé [14] support this way of using OWL quite well.

State-of-the-art description logic (DL) inference systems such as Racer allow for interpreting OWL ontology documents as T-boxes and A-boxes [8]. Racer accepts the OWL DL subset [17] (with the additional restriction of approximated reasoning

[*]OntoXpl's download page: http://www.cs.concordia.ca/ying_lu/

for so-called nominals and no full number restrictions for datatype properties). Descriptions of individuals are represented as A-boxes by the Racer System (for details see the Racer User's Guide [7]). Viewing the RDF part of OWL DL documents as A-boxes provides for query languages supported by DL systems. Furthermore, graphical interfaces for description logic inference systems can be used to inspect OWL ontologies.

User interfaces are very important for practical work with description logic inference systems. An increasing number of graphical interfaces are available for existing DL systems. One class of interfaces consists of ontology editors such as OilEd [2] and Protégé [14]. With these editors ontologies can be interactively built and they can be stored, for example, as OWL documents. In addition, the editors can be used to develop RDF documents for describing information about individuals with respect to OWL ontologies. Applications using these OWL documents require an inference engine that supports reasoning about individuals. Indeed, OilEd and Protégé can be configured to use Racer [6] as an inference engine for classifying ontologies and for answering simple queries about individuals.

The second class of interfaces offers browsing and visualization capabilities. Rice [13] supports the input of textual queries and displays the concept/class hierarchy of T-boxes as outline views as well as the relational structure of A-boxes as directed graphs. The outline view of classes is usually also supported by ontology editors but Rice additionally supports the visualization of A-boxes. Other OWL/RDF visualization tools or editors with visualization capabilities are, e.g., KAON [15], OntoEdit [16], and OntoTrack [11].

The OWL ontology explorer OntoXpl presented in this paper is intended to complement existing ontology editors and visualization tools. It is completely based on OWL and offers a large variety of information queries. Three potential user groups are targeted by OntoXpl's design: (i) users with a limited background of ontologies and OWL; (ii) ontology developers that are OWL experts; (iii) users interested in understanding and reusing existing ontologies. OntoXpl is available as a web server based on the tomcat architecture. Standard HTML browsers can be used to interact with OntoXpl. Its interface makes heavy use of Racer's extensive query interface in order to support users when exploring OWL ontologies. The following sections give a brief tour on using OntoXpl and explain its rationale in more detail. Afterwards OntoXpl is compared with related work. This paper concludes with an outlook to possible future work.

## 2 OntoXpl's main user interface

OntoXpl's design is influenced by OWL (and its foundation on DLs).[1] Therefore, it focuses on the three main language elements of OWL, classes/concepts, roles/properties, and nominals/individuals.

The main command pane of OntoXpl is shown in Figure 1. The filename of the OWL ontology currently loaded into OntoXpl and Racer is shown with a summary of the number of contained concept and role names (see also Section 3 for an

[1]The DL and OWL vocabulary is used interchangeably in this paper.

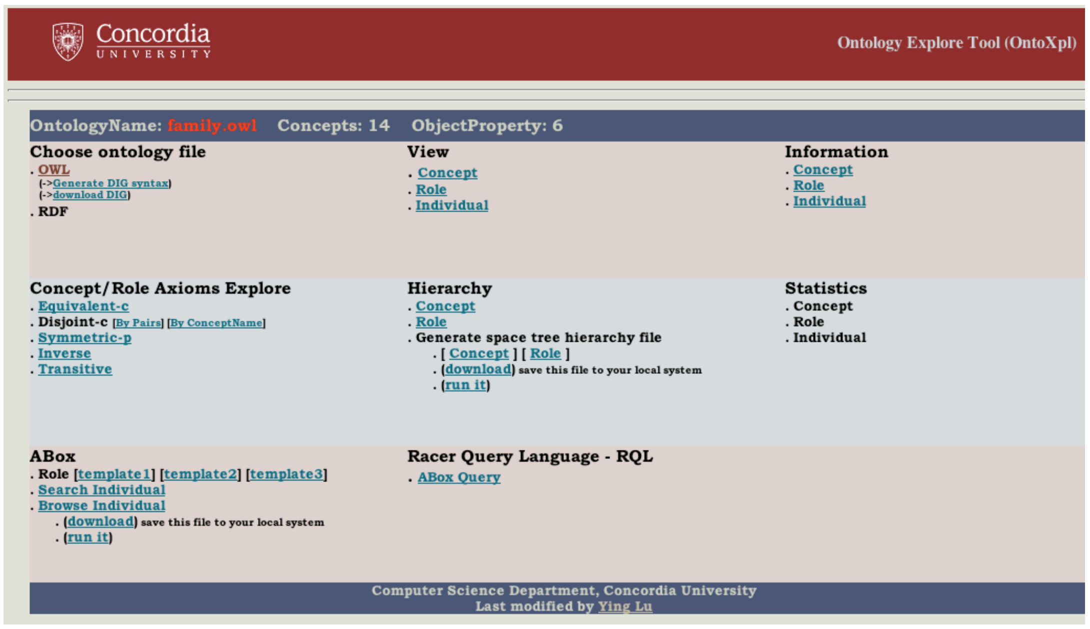


Figure 1: OntoXpl's main command pane.

explanation of the example knowledge base). OntoXpl's interface offers eight principal browsing categories (from left to right and top to bottom): file selector, "natural language" description, structural information, exploration of concept/property axioms, inspection of concept and role hierarchies, view of statistical information (not yet implemented), inspection of A-box graph structures, and the interactive use of Racer's query language nRQL. In the following the seven implemented categories are described.

Figure 2 shows a zoom of the first (horizontal) command pane. The left group of commands is used to load an OWL file and generate a DIG representation of the loaded OWL file. The middle group of commands applies to concepts, roles, and individuals. These commands result in displaying the OWL source code (e.g., see Figure 8) together with a "natural language" description (e.g., see Figure 7). The "natural language" (NL) description is based on the DL notation and tries to describe the selected item w.r.t. this notation. These NL descriptions are intended for users with a limited background on DL and OWL. The information views of concepts (e.g., see the window at the left-hand side of Figure 9), roles (e.g., see Figure 10), and individuals (e.g., see the two windows at the right-hand side of Figure 9) use Racer's query interface to display their (inferred) characteristics. Concepts are described by (i) their relative position in the classification hierarchy (e.g., parent, children), (ii) the roles occurring in the concept declarations, and (iii) the individuals that are instances of this concept. By analogy, a role is similarly described but in addition to its position in the role hierarchy, the concepts are listed that use this role. An individual is described by (i) its most specific concept names (so-called types) of which it is an

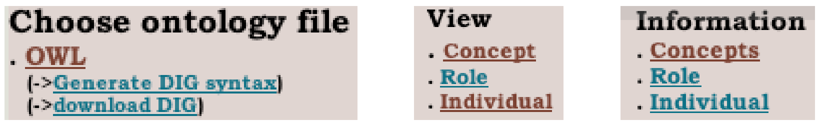


Figure 2: Zoom of the upper three command menus (from left to right): file selection, OWL / natural language views, information page views.

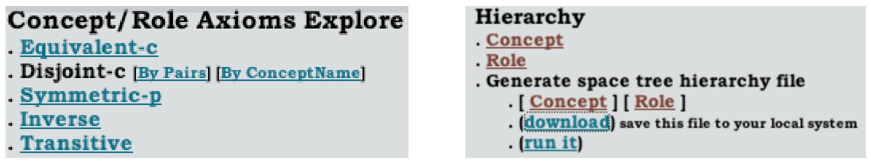

Concept/Role Axioms Explore
. Equivalent-c
. Disjoint-c [By Pairs] [By ConceptName]
. Symmetric-p
. Inverse
. Transitive
Hierarchy
. Concept
. Role
. Generate space tree hierarchy file
. [ Concept ] [ Role ]
. (download) save this file to your local system
. (run it)

Figure 3: Zoom of the middle two command menus (from left to right): explore concept/property characteristics, show concept/role hierarchies.

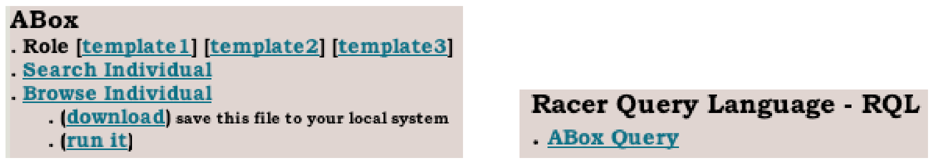

ABox
. Role [template1] [template2] [template3]
. Search Individual
. Browse Individual
. (download) save this file to your local system
. (run it)
Racer Query Language - RQL
. ABox Query

Figure 4: Zoom of the bottom two command menus (from left to right): A-box command menu, nRQL Racer Query Language.

instance, (ii) other individuals that are instances of concepts (parents, children, etc) related to its types.

The two implemented command groups from the middle pane are shown in Figure 3. The command group displayed on the left allows one to query about equivalent or disjoint concept names and symmetric, inverse, and transitive roles. The other group is concerned with concept and role hierarchies. There exist two principal services: (i) one can browse the concept or roles hierarchies in an outline view; (ii) a data file for the SpaceTree tool [5] is generated such that the taxonomies can be graphically inspected.[2]

The last two command groups from the bottom pane are shown in Figure 4. They are dedicated to explore A-boxes. The first command group has several search forms to retrieve individuals and their known relationships with other individuals, to browse relationships in an outline view or inspect the A-box structure with SpaceTree. The second command group allows users to query A-boxes with Racer's query language nRQL [9].

[2]The hierarchy is shown as a pure tree, i.e., edges to more than one superclass are ignored.

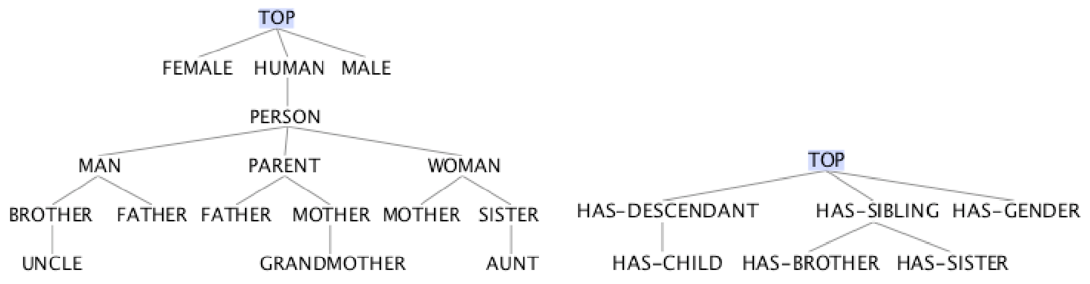


Figure 5: Class (left) and property hierarchy (right) of the "family" KB.

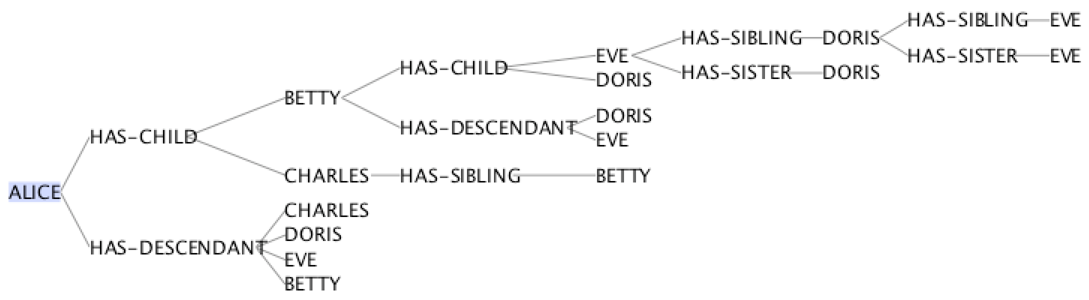
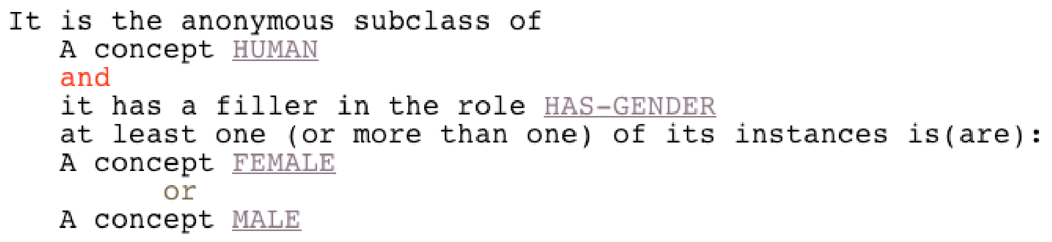


Figure 6: Graph of the A-box relationships.



```
It is the anonymous subclass of
   A concept HUMAN
   and
   it has a filler in the role HAS-GENDER
   at least one (or more than one) of its instances is(are):
   A concept FEMALE
         or
   A concept MALE
```

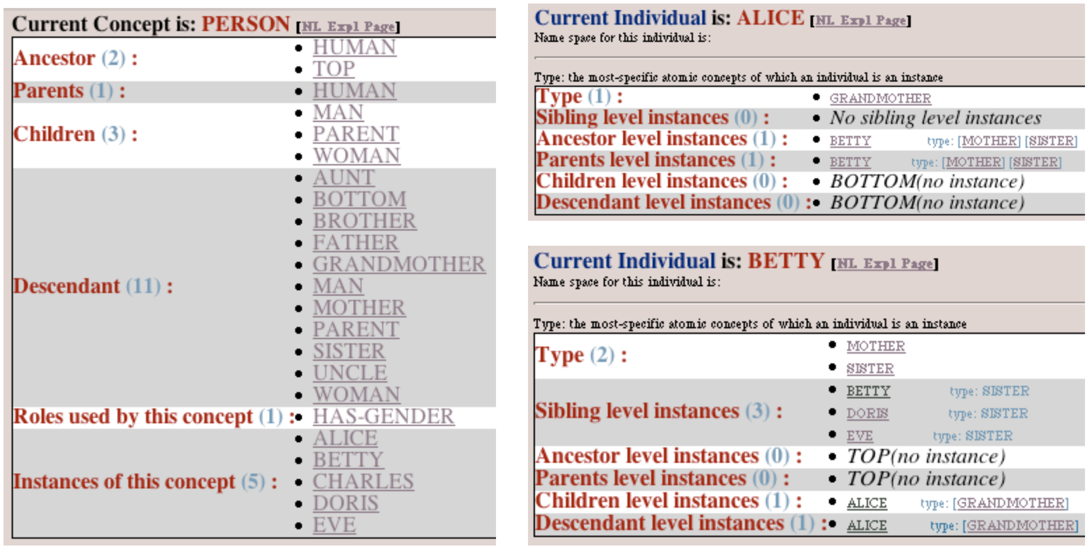
Figure 7: "Natural Language" description of class PERSON.

## 3 Example scenario

The capabilities of OntoXpl are best explored interactively. However, in this section we try to briefly illustrate some of its main features. Let us assume that OntoXpl is used to explore an ontology file called "family.owl" describing knowledge about family members (e.g., mother, aunt) and their relationships (e.g., has-child, has-sibling). The structure of the corresponding class and role hierarchies is shown in Figure 5 and the structure of the A-box in Figure 6. From the T-box graph a user might be interested in the class PERSON and selects this class for further inspection. Figure 7 shows a "natural language" (NL) description of this class (the underlined names link to the

```
(rdfs:subClassOf)
        (owl:Class)
                (owl:intersectionOf rdf:parseType="Collection")
                        (owl:Class rdf:about="HUMAN")
                        (/owl:Class)
                        (owl:Restriction)
                                (owl:onProperty rdf:resource="HAS-GENDER")
                                (/owl:onProperty)
                                (owl:someValuesFrom)
                                        (owl:Class)
                                                (owl:unionOf rdf:parseType="Collection")
                                                        (owl:Class rdf:about="FEMALE")
                                                        (/owl:Class)
                                                        (owl:Class rdf:about="MALE")
                                                        (/owl:Class)
                                                (/owl:unionOf)
                                        (/owl:Class)
                                (/owl:someValuesFrom)
                        (/owl:Restriction)
                (/owl:intersectionOf)
        (/owl:Class)
(/rdfs:subClassOf)
```

Figure 8: OWL specification of class PERSON.



**Current Concept is: PERSON** [NL Expl Page]

| | |
|---|---|
| Ancestor (2) : | • HUMAN<br>• TOP |
| Parents (1) : | • HUMAN |
| Children (3) : | • MAN<br>• PARENT<br>• WOMAN |
| Descendant (11) : | • AUNT<br>• BOTTOM<br>• BROTHER<br>• FATHER<br>• GRANDMOTHER<br>• MAN<br>• MOTHER<br>• PARENT<br>• SISTER<br>• UNCLE<br>• WOMAN |
| Roles used by this concept (1) : | • HAS-GENDER |
| Instances of this concept (5) : | • ALICE<br>• BETTY<br>• CHARLES<br>• DORIS<br>• EVE |

**Current Individual is: ALICE** [NL Expl Page]

Name space for this individual is:

Type: the most-specific atomic concepts of which an individual is an instance

| | | |
|---|---|---|
| Type (1) : | • GRANDMOTHER | |
| Sibling level instances (0) : | • *No sibling level instances* | |
| Ancestor level instances (1) : | • BETTY | type: [MOTHER] [SISTER] |
| Parents level instances (1) : | • BETTY | type: [MOTHER] [SISTER] |
| Children level instances (0) : | • *BOTTOM(no instance)* | |
| Descendant level instances (0) : | • *BOTTOM(no instance)* | |

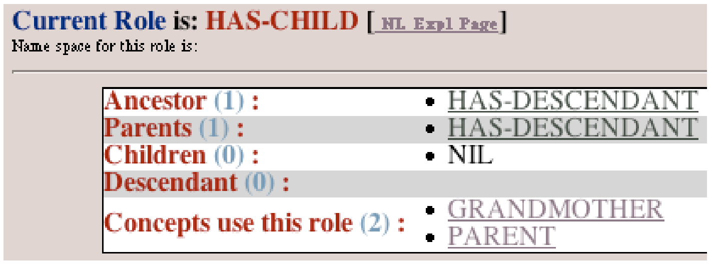
**Current Individual is: BETTY** [NL Expl Page]

Name space for this individual is:

Type: the most-specific atomic concepts of which an individual is an instance

| | | |
|---|---|---|
| Type (2) : | • MOTHER<br>• SISTER | |
| Sibling level instances (3) : | • BETTY<br>• DORIS<br>• EVE | type: SISTER<br>type: SISTER<br>type: SISTER |
| Ancestor level instances (0) : | • *TOP(no instance)* | |
| Parents level instances (0) : | • *TOP(no instance)* | |
| Children level instances (1) : | • ALICE | type: [GRANDMOTHER] |
| Descendant level instances (1) : | • ALICE | type: [GRANDMOTHER] |

Figure 9: Taxonomic information about the class PERSON (left) and the individuals ALICE (top-right) and BETTY (bottom-right).

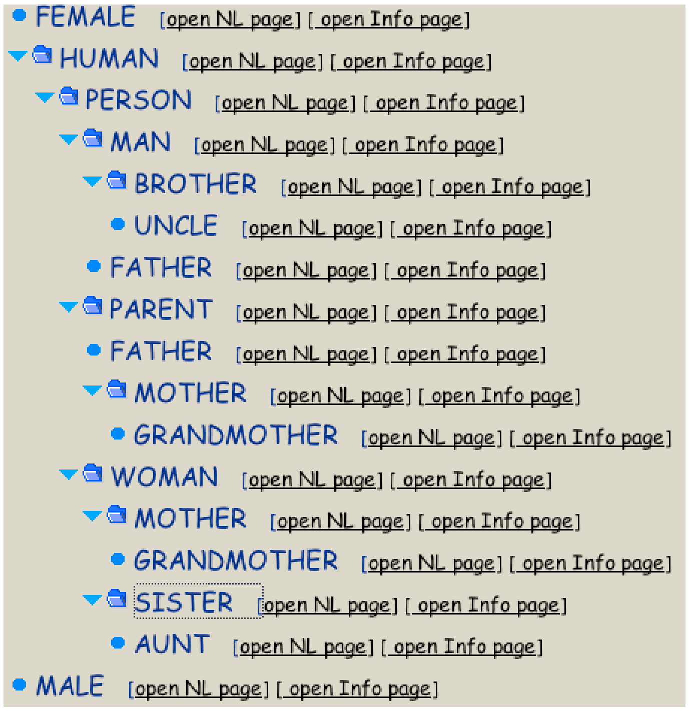


**Current Role is: HAS-CHILD** [ NL Expl Page ]

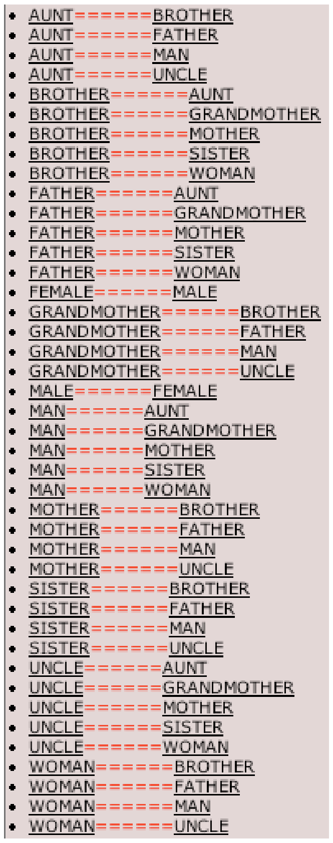
Name space for this role is:

| | |
|---|---|
| Ancestor (1) : | • HAS-DESCENDANT |
| Parents (1) : | • HAS-DESCENDANT |
| Children (0) : | • NIL |
| Descendant (0) : | |
| Concepts use this role (2) : | • GRANDMOTHER<br>• PARENT |

Figure 10: Taxonomic information about the role HAS-CHILD.



```
● FEMALE [open NL page] [ open Info page]
▼ HUMAN [open NL page] [ open Info page]
  ▼ PERSON [open NL page] [ open Info page]
    ▼ MAN [open NL page] [ open Info page]
      ▼ BROTHER [open NL page] [ open Info page]
        ● UNCLE [open NL page] [ open Info page]
      ● FATHER [open NL page] [ open Info page]
    ▼ PARENT [open NL page] [ open Info page]
      ● FATHER [open NL page] [ open Info page]
      ▼ MOTHER [open NL page] [ open Info page]
        ● GRANDMOTHER [open NL page] [ open Info page]
    ▼ WOMAN [open NL page] [ open Info page]
      ▼ MOTHER [open NL page] [ open Info page]
        ● GRANDMOTHER [open NL page] [ open Info page]
      ▼ SISTER [open NL page] [ open Info page]
        ● AUNT [open NL page] [ open Info page]
● MALE [open NL page] [ open Info page]
```



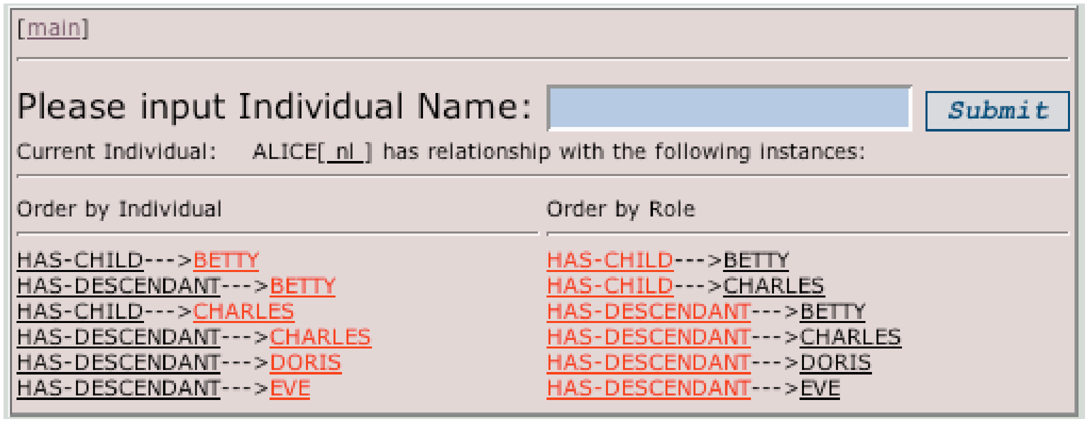
- AUNT======BROTHER
- AUNT======FATHER
- AUNT======MAN
- AUNT======UNCLE
- BROTHER======AUNT
- BROTHER======GRANDMOTHER
- BROTHER======MOTHER
- BROTHER======SISTER
- BROTHER======WOMAN
- FATHER======AUNT
- FATHER======GRANDMOTHER
- FATHER======MOTHER
- FATHER======SISTER
- FATHER======WOMAN
- FEMALE======MALE
- GRANDMOTHER======BROTHER
- GRANDMOTHER======FATHER
- GRANDMOTHER======MAN
- GRANDMOTHER======UNCLE
- MALE======FEMALE
- MAN======AUNT
- MAN======GRANDMOTHER
- MAN======MOTHER
- MAN======SISTER
- MAN======WOMAN
- MOTHER======BROTHER
- MOTHER======FATHER
- MOTHER======MAN
- MOTHER======UNCLE
- SISTER======BROTHER
- SISTER======FATHER
- SISTER======MAN
- SISTER======UNCLE
- UNCLE======AUNT
- UNCLE======GRANDMOTHER
- UNCLE======MOTHER
- UNCLE======SISTER
- UNCLE======WOMAN
- WOMAN======BROTHER
- WOMAN======FATHER
- WOMAN======MAN
- WOMAN======UNCLE

Figure 11: Outline view of class hierarchy (left) and all pairs of disjoint concept names (right).

corresponding NL views) while Figure 8 displays its OWL specification (using the XML syntax). The NL and OWL views are directly linked with the corresponding taxonomic views. Figure 9 displays the taxonomic information about the class PERSON retrieved from RACER. It lists ancestors, parents, children, and descendants of PERSON. It also shows the role names used in this class specification and the individuals which are instances of PERSON.

A user might be interested in the individual ALICE. Its taxonomic information is shown in Figure 9, e.g., ALICE is an instance of GRANDMOTHER. This view also lists instances of concepts that are ancestors, parents, siblings, descendants, or children of ALICE's most-specific subsumers (GRANDMOTHER). For instance, BETTY is an instance of these parent classes. The corresponding information about BETTY is shown in Figure 9. Figure 10 shows the taxonomic information about the role HAS-CHILD. The display of inferred information in these windows is intended to help users better understand the structure of the T-boxes and A-boxes.

In contrast to the hierarchical views displayed by using the SpaceTree tool, ONTOXPL also offers its own outline views for concept and role hierarchies as well as

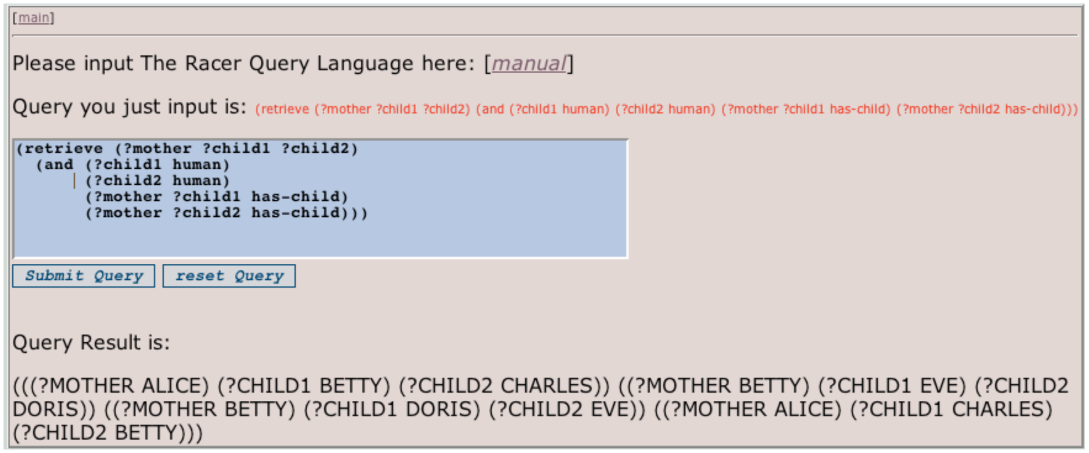

[main]

Please input Individual Name: Submit

Current Individual: ALICE[ nl ] has relationship with the following instances:

| Order by Individual | Order by Role |
|---|---|
| HAS-CHILD--->BETTY | HAS-CHILD--->BETTY |
| HAS-DESCENDANT--->BETTY | HAS-CHILD--->CHARLES |
| HAS-CHILD--->CHARLES | HAS-DESCENDANT--->BETTY |
| HAS-DESCENDANT--->CHARLES | HAS-DESCENDANT--->CHARLES |
| HAS-DESCENDANT--->DORIS | HAS-DESCENDANT--->DORIS |
| HAS-DESCENDANT--->EVE | HAS-DESCENDANT--->EVE |

Figure 12: Asserted and inferred role fillers of ALICE.


[main]

Please input The Racer Query Language here: [*manual*]

Query you just input is: (retrieve (?mother ?child1 ?child2) (and (?child1 human) (?child2 human) (?mother ?child1 has-child) (?mother ?child2 has-child)))

```
(retrieve (?mother ?child1 ?child2)
  (and (?child1 human)
       (?child2 human)
       (?mother ?child1 has-child)
       (?mother ?child2 has-child)))
```

Submit Query reset Query

Query Result is:

(((?MOTHER ALICE) (?CHILD1 BETTY) (?CHILD2 CHARLES)) ((?MOTHER BETTY) (?CHILD1 EVE) (?CHILD2 DORIS)) ((?MOTHER BETTY) (?CHILD1 DORIS) (?CHILD2 EVE)) ((?MOTHER ALICE) (?CHILD1 CHARLES) (?CHILD2 BETTY)))

Figure 13: Example nRQL query and its result.

A-box structures. The left-hand side of Figure 11 shows the complete unfolded hierarchy using an outline view. The disadvantage of this type of view is the repeated occurrence of classes (or subtrees) that have more than one parent (e.g., FATHER, MOTHER). The right-hand side of Figure 11 displays all pairs of disjoint concept names.

Figure 12 displays information about the asserted and inferred role fillers of ALICE (ordered by individual or role name). Figure 13 shows a complex nRQL [9] query which searches for children having a common mother. The dialog box displays the input query and its returned result in a Lisp-like notation.

## 4 Discussion

Currently there do not exist many stable and usable ontology visualization or exploration tools (and even editors). The lack of suitable tools and their shortcomings were one of the major motivations to design and implement OntoXpl. The motivation for OntoXpl's web server based architecture was the ease of use with standard HTML browsers and the simple adaptation to multi-user environments. To the best of our knowledge OntoXpl is currently the only ontology exploration tool that is fully targeted to OWL and relies on Racer's deductive capabilities for offering users better exploration capabilities. A detailed description of OntoXpl and its architecture as well as a comparison with related work can be found in [12].

Various features of OntoXpl are also (partly) supported by ontology editors such as Protégé [14] and OilEd [2] or OWL/RDF visualization tools or editors with visualization capabilities such as KAON [15], OntoEdit [16], and OntoTrack [11]. For instance, Protégé also offers users a high-level (DL-like) description of the definition of concept names if the mouse pointer is moved over these names. However, this does not seem to be very suitable for longer concept definitions and does not support the inspection of the occurring OWL elements via hyperlinks. OntoXpl's "NL description" seems to be more readable and carefully supports the inspection of mentioned entities via hyperlinks. Rice [13] offers visualization facilities for A-boxes where the complete graph structure of A-boxes is displayed. OntoXpl is restricted to a tree-like approximation due to the underlying SpaceTree tool [5] but it works better for larger A-boxes.

In our experience, OntoXpl's cross-referencing capabilities for hyperlinked concept, role, and individual names help users comprehend unknown ontologies faster than with the support offered by traditional editors.

## 5 Conclusion

In this paper we briefly introduced OntoXpl, a first step toward an OWL ontology exploration tool. OntoXpl is intended to complement ontology editors or other ontology visualization tools. A recently conducted informal experiment, where about 40 students had to design and implement 15 different OWL ontologies with a size of several hundred concept names, demonstrated that OntoXpl provides helpful information about ontologies that is otherwise not as easily available in ontology editors such as Protégé or OilEd. The implementation of the statistics command group is underway. It is also planned to integrate query results from nRQL such that individuals names are recognized as hyperlinks. Another important issue is the optimization of OntoXpl performance for larger ontologies containing thousands of concept names.

## References

[1] F. Baader, I. Horrocks, and U. Sattler. Description logics as ontology languages for the semantic web. In D. Hutter and W. Stephan, editors, *Festschrift in honor of Jörg Siekmann*. LNAI. Springer-Verlag, 2003.

[2] S. Bechhofer, I. Horrocks, and C. Goble. OilEd: a reason-able ontology editor for the semantic web. In *Proceedings of KI2001, Joint German/Austrian conference on Artificial Intelligence, September 19-21, Vienna*. LNAI Vol. 2174, Springer-Verlag, 2001.

[3] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, May 2001.

[4] D. Brickley and R.V. Guha. RDF vocabulary description language 1.0: RDF Schema, http://www.w3.org/tr/2002/wd-rdf-schema-20020430/, 2002.

[5] J. Grosjean, C. Plaisant, and B. Bederson. SpaceTree: Supporting exploration in large node link tree, design evolution and empirical evaluation. In *Procedings of IEEE Symposium on Information Visualization*, pages 57–64, Boston, USA, October 2002.

[6] V. Haarslev and R. Möller. RACER system description. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2001)*, 2001.

[7] V. Haarslev and R. Möller. The Racer user's guide and reference manual, 2003.

[8] V. Haarslev and R. Möller. Optimization techniques for retrieving resources described in OWL/RDF documents: First results. In *Proceedings of the Ninth International Conference on the Principles of Knowledge Representation and Reasoning (KR2004)*, June 2004.

[9] V. Haarslev, R. Möller, R. Van Der Straeten, and M. Wessel. Extended query facilities for Racer and an application to software-engineering problems. In *Proceedings of the International Workshop on Description Logics (DL-2004), Whistler, BC, Canada*, June 2004.

[10] O. Lassila and R.R. Swick. Resource description framework (RDF) model and syntax specification. recommendation, W3C, february 1999. http://www.w3.org/tr/1999/rec-rdf-syntax-19990222, 1999.

[11] T. Liebig and O. Noppens. OntoTrack: Fast browsing and easy editing of large ontologies. In *Proceedings of The Second International Workshop on Evaluation of Ontology-based Tools (EON2003), located at ISWC03*, Sanibel Island, USA, October 2003.

[12] Y. Lu. Exploration of OWL ontologies. Master's thesis, Department of Computer Science, Concordia University, Montreal, 2004 (in preparation).

[13] R. Möller, R. Cornet, and V. Haarslev. Graphical interfaces for Racer: querying DAML+OIL and RDF documents. In *Proc. International Workshop on Description Logics – DL'03*, 2003.

[14] N. F. Noy, M. Sintek, 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, 2001.

[15] D. Oberle, R. Volz, B. Motik, and S. Staab. An extensible ontology software environment. In *Handbook on Ontologies*, International Handbooks on Information Systems, chapter III, pages 311–333. Steffen Staab and Rudi Studer, Eds., Springer, 2004.

[16] Y. Sure, J. Angele, and S. Staab. OntoEdit: multifaceted inferencing for ontology engineering. *Journal on Data Semantics*, 2800/2003:128–152, 2003.

[17] F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. OWL web ontology language reference, http://www.w3.org/tr/owl-guide/, 2003.