Hypermedia Inspired Ontology Engineering Environment: SWOOP

Aditya Kalyanpur[1], Evren Sirin[2], Bijan Parsia[3] and James Hendler[4]
Department of Computer Science {[1, 2, 4]}, MIND Lab [3,4],
University of Maryland, College Park
{aditya[1], evren[2], hendler[4]}@cs.umd.edu, bparsia@isr.umd.edu[3]

Abstract

In this paper, we present the design and architecture of a hypermedia inspired ontology engineering environment - **SWOOP**. With its web-metaphor, adherence to OWL recommendations, fluid ontology manipulation interface, and easy extensibility it acts as a useful and efficient web ontology development tool.

1 Introduction

Most existing ontology development toolkits such as Protégé [Stanford, 2000], Oiled [Bechhofer et al, 2001], OntoEdit [Sure et al, 2001], WebODE [Arpírez et al, 2001], provide an integrated environment to build and edit ontologies, check for errors and inconsistencies (using a reasoner), browse multiple ontologies, and share and reuse existing data by establishing mappings among different ontological entities. However, their UI design (look & feel) and usage style are inspired by traditional KR-based paradigms, whose constrained and methodical framework have steep-learning curves, making it cumbersome to use for the average web user. On the other hand, consider a hypermedia inspired ontology editor that employs a web-based metaphor for its design and usage. As argued in [Kalyanpur et al, 2004], such a tool would be more effective (in terms of acceptance and use) for the average web user by presenting a simpler, consistent and familiar framework for dealing with entities on the Semantic Web. Based on this hypothesis, we present our ontology editor - SWOOP, meant for rapid and easy browsing and development of web ontologies.

1.1 SWOOP

As noted earlier, SWOOP is a simple, scalable, hypermedia-inspired OWL ontology browser and editor. It uses the URIs which are the primary identifiers for ontologies, classes, properties, and individuals to support hypertext*esque* navigation through, and between, ontologies. The correlate to a Web page is an OWL Ontology, with the contained entities (classes / properties/individuals) being analogous to HTML anchors embedded in the page. The ontological "page" can be rendered in various formats ranging from a concise logic based view to a more verbose RDF/XML representation. Other familiar web-browser look&feel features include an address bar to load ontological entities, history buttons and bookmarks. Moreover, SWOOP has been designed in-keeping with the W3C OWL recommendations, i.e. it contains OWL species validation, offers various OWL Presentation Syntax views (Abstract Syntax, N3 etc), has reasoning support (OWL Inference Engine), and provides a Multiple Ontology environment, whereby entities and relationships across various ontologies can be compared, edited and merged seamlessly. Finally, it also has a Wiki-style inline editing facility with a change log and an undo/redo option. In the next section, we briefly discuss its design goals and distinguishing features using screenshots as our primary basis.

2 Design Goals and Features

2.1 Familiar user-friendly Weblike Look & Feel

2.1.1 Browser components (address bar, hyperlinked navigation, history buttons, bookmarks, see **Fig. 1**)

2.1.2 Inline editing (using different color codes and font styles to emphasize ontology changes, see **Fig. 1**)

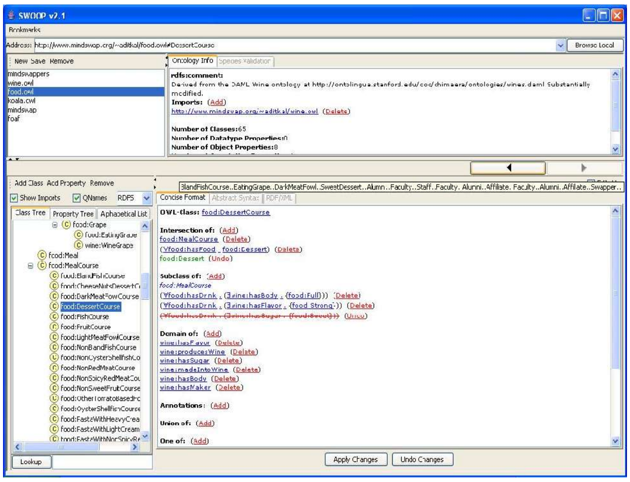

Figure 1: SWOOP User Interface

2.2 Ontology Annotation and Versioning

2.2.1 Collaborative Annotation Support using Annotea: An Annotea [Kahan et al, 2001] plug-in in SWOOP allows users to write and share annotations on any ontological entity (class/ property/ individual). This is an useful utility in a highly unconstrained environment like the Semantic Web where users value a variety of perspectives, descriptions, and explanations on an informational web resource (see **Fig. 2**).

2.2.2 Enforcing Ontology Changes Dynamically: A side-effect of the above, any change made to an ontology or its components can be annotated as well (using a default schema provided), and the change actions along with the annotations can be stored in a public server. Different SWOOP users can then subscribe to the server, download annotated changes for a given ontology, and maintain different versions of the same ontology by applying changes selectively.

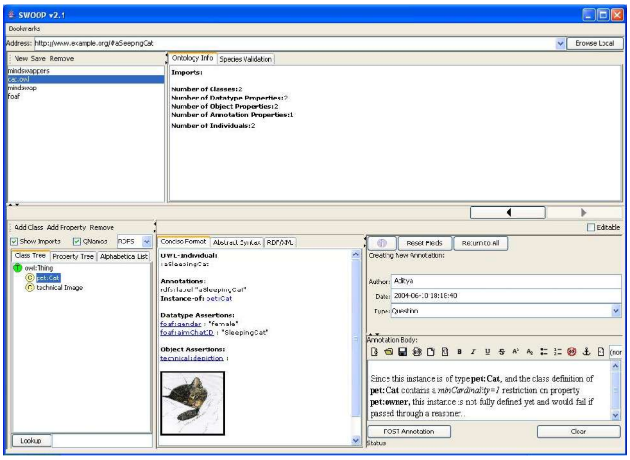

Figure 2: Annotea support in SWOOP

2.3 Search and Reuse of Heterogeneous Ontological Data

2.3.1 Concept Search: SWOOP employs an ontology search algorithm that combines keywords with DL-based constructs to find related concepts in existing ontologies (algorithm described in [Kalyanpur et al, 2004])

2.3.2 Comparing and directly linking entities from multiple ontologies: SWOOP also has a provision to compare similar terms present in different ontologies against their DL-based definitions, associated properties and sample-instances

2.3.3 Complex Mappings and Reuse: Having found related concepts/properties in external ontologies, SWOOP assists the user in either linking to the data (with or without importing the entire external ontology, using a novel ontology mapping approach developed in the DL community [Cuenca-Grau et al, 2004]) or borrowing a specific subset of an external ontology (using a copy-paste mechanism, explained in [Kalyanpur et al, 2004]).

2.4 Other useful Plug-ins

2.4.1 Semantic Markup support: Multimedia markup Plug-ins such as those that facilitate image and video annotations fit in nicely with the fluid hypermedia-based SWOOP UI (see **Fig. 2**)

3 Architecture

SWOOP is based on the conventional **Model-View Controller (MVC)** paradigm and the basic components are shown in **Fig. 3.** It uses the **WonderWeb OWL API**, a high level programmatic interface written in Java to parse and manipulate OWL Ontologies [Bechhofer et al, 2003].

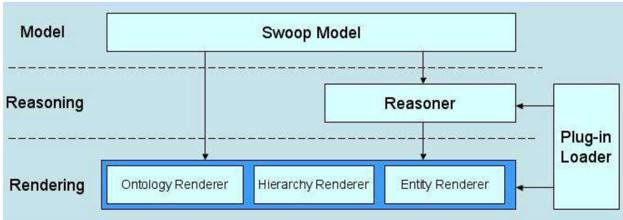

Figure 3: SWOOP Architecture

4 Conclusion and Future Work

This paper outlines our contribution in building a hypermedia inspired ontology editing tool – SWOOP. However, it still represents work in progress. Some of the solutions proposed in the paper need to be elaborated upon, implemented and optimized in SWOOP. Moreover, a formal evaluation of the features needs to be done by performing usability studies and comparing it against existing ontology engineering tools.

References

[Stanford., 2000] Using Protégé-2000 to Edit RDF. Technical Report. Stanford University.

[Bechhofer et al, 2001] OilEd: a Reasonable Ontology Editor for the Semantic Web. Proceedings of KI2001,

[Sure et al, 2002] OntoEdit: Collaborative Ontology Development for the Semantic Web. International Semantic Web Conference (ISWC02). Sardinia. Italy.

[Arpírez et al, 2001] WebODE: a scalable ontological engineering workbench. First International Conference on Knowledge Capture (K-CAP 2001), Canada

[Kalyanpur et al, 2004] Lifecycle of a Casual Web Ontology Development Process, Proc. of the WWW2004 WE-SW Workshop, NYC, USA

[Kahan et al, 2001] Annotea: An Open RDF Infrastructure for Shared Web Annotations, in Proc. of the WWW10 International Conference, Hong Kong

[Cuenca-Grau et al, 2004] Working with Multiple Ontologies on the Semantic Web. To appear in Proceedings of the ISWC 2004, Japan

[Bechhofer et al, 2003] Cooking the Semantic Web with the OWL API. International Semantic Web Conference (ISWC03). Sanibel Island, USA