Swoop: A Web Ontology Engineering Framework

Aditya Kalyanpur, Bijan Parsia, Evren Sirin, Bernardo Cuenca Grau, James Hendler

aditya@cs.umd.edu, bparsia@isr.umd.edu, evren@cs.umd.edu

bernardo@mindlab.umd.edu, hendler@cs.umd.edu

Maryland Information and Network Dynamics Laboratory,
University of Maryland,
College Park, MD 20740, USA

Abstract

Swoop is a hypermedia-inspired ontology browser and editor based on OWL, the Web Ontology Language. A diverse array of ontology related tasks can be performed in Swoop ranging from collaborative annotation and data markup to ontology refactoring and debugging. This makes Swoop accessible to both, novice users interested in casual ontology building and use, and expert users interested in robust OWL ontology modeling and analysis.

1 Introduction

Swoop is a hypermedia-inspired Ontology Browser and Editor based on OWL, the Web Ontology Language. Swoop takes the standard Web browser as the UI paradigm, believing that URIs are central to the understanding and construction of OWL Ontologies. The familiar look and feel of a browser emphasized by the address bar and history buttons, navigation side bar, bookmarks, hypertextual navigation etc are all supported for web ontologies, corresponding with the mental model people have of URI-based web tools based on their current Web browsers.

All design decisions are in keeping with the OWL nature and specifications. Thus, multiple ontologies are supported easily, various OWL presentation syntax are used to render ontologies, open-world semantics are assumed while editing and OWL reasoners can be integrated for consistency checking. A key point in our work is that the hypermedia basis of the UI is exposed in virtually every aspect of ontology engineering — easy navigation of OWL entities, comparing and editing related entities, search and cross referencing, multimedia support for annotation, etc. — thus allowing ontology developers to think of OWL as just another Web format, and thereby take advantage of its Web-based features.

2 Basic Features

Swoop functionality is characterized by the following basic features (for an elaborate discussion of the features see [1]):

- *Multiple Ontology Browsing and Editing:* Swoop has a variety of mechanisms for pulling in different Web ontologies into its model - using bookmarks; loading via the address bar; during navigation across ontologies etc. Additionally, ontology browsing and editing are done in a single pane, which helps to maintain context.
- *Renderer Plugins for OWL Presentation Syntaxes:* Swoop bundles in six renderers; two Ontology Renderers–Information and Species Validation; and four Entity Renderers - Concise Format, OWL Abstract Syntax, Turtle and RDF/XML. By supporting different presentation syntaxes, accessibility is enhanced tremendously.
- *Reasoner plugins in Swoop:* Swoop contains two additional reasoners (besides the basic Reasoner that simply uses the asserted structure of the ontology): RDFS-like and Pellet [2]. While the former is a lightweight reasoner based on RDFS semantics, the latter, Pellet, is a powerful description logic tableaux reasoner. The reasoners provide a tradeoff between speed and quality of inference results, e.g., the RDFS-like reasoner, while much faster than Pellet in execution, is unsound (results maybe inaccurate if the ontology is inconsistent) and incomplete (does not list all possible inferences). Yet, in most cases, it provides interesting and useful results for ontology authors, and moreover, the reasoners can be used in conjunction to analyze the ontology quickly while editing it.
- *Semantic Search:* Swoop takes inspiration from the hyperlink based search and cross-referencing utility present in a programming IDE. For example, a non-standard search feature in Swoop is "Show References", which lists all references of a single OWL entity (concept/property/individual) in local or external ontological definitions. This feature can help understand usage of an entity in a specific context and is especially useful in ontology debugging where non-local interactions can lead to errors of a single concept.
- *Ontology Versioning:* Swoop supports ad hoc undo/redo of changes (with logging) coupled with the ability to checkpoint and archive different ontology versions. Each change set or checkpoint can be saved at three different granularity levels - entity, ontology, workspace, which basically specify its scope. While the change logs can be used to explicitly track the evolution of an ontology, checkpoints allows the user to switch between versions directly exploring different modeling alterna-

tives. Swoop also has the option to save checkpoints automatically, i.e., during specific tasks such as loading a new ontology, applying changes, removing or renaming an entity etc. Note that each time a checkpoint is saved, a snapshot of the entity definition is cached as well, and can be used to preview a checkpoint before reverting back to it.

3 Advanced Features

Additionally, Swoop has the following advanced features, which represent work in progress:

- *Resource Holder (Comparator/Mapping utility):* In Swoop, there is a provision to store and compare OWL entities during an extended search and browsing process via the Resource Holder panel. This common placeholder acts as an excellent platform for performing interesting engineering tasks such as comparing differences in definitions of a set of entities; determining semantic mappings between a specific pair of entities or simply storing entities for reusing in another ontology.
- *Automatic Partitioning of OWL Ontologies (using E-Connections):* Swoop has a provision for automatically partitioning OWL Ontologies by transforming them into an E-connection. For more details on the theory and significance of E-connections, their use in the context of OWL Ontologies see [3]
- *Ontology Debugging and Repair:* Swoop uses two techniques for ontology debugging and repair: glass box and black box. In glass box techniques, information from the internals of the reasoner is extracted and presented to the user (typically used to pinpoint the type of clash/contradiction and axioms leading to the clash [4]). In black box techniques, the reasoner is used as an oracle for a certain set of questions e.g., the standard description logic inferences (subsumption, satisfiability, etc.) and the asserted structure of the ontology is used to help isolate the source of the problems (can be used to find dependencies between unsatisfiable classes).
- *Annotea Client in Swoop: Collaborative Annotations and Change Sets* The Annotea protocol allows for publishing and finding out-of-band annotations on arbitrary web resources. Annotea support in Swoop is provided via a simple plug in that uses the default Annotea RDF schema to specify annotations. Any public Annotea Server can then be used to publish and distribute the annotations created in Swoop. In addition to annotations, users can attach and share Ontology Change sets that are created using Swoop.

4 Conclusion

In designing Swoop, we take the familiar Web browser as our User Interface (UI) paradigm, focusing on a simple, intuitive and fluid ontology development interface. The primary goal has been to design a tool based on the successful architecture of the Web itself, i.e., its *open* (ad hoc local and remote multiple ontology support, various OWL Presentation syntax, extensible plugin architecture), *scalable* (e.g., ontologies as large as NCI with over 27000 classes can be loaded in Swoop is under 2 minutes), and *distributed* (e.g., collaborative annotation support via Annotea). By doing the above, and remaining true to the OWL specifications, we allow the Swoop user to take advantage of the Web-based features of OWL.

Our secondary goal has been on overcoming the drawbacks, if any, of traditional ontology development tasks as applied to OWL. For instance, we have presented a partitioning approach for OWL ontologies to solve the *partial owl:imports* problem while facilitating ontology reuse. We have also presented work on guiding the user through the process of ontology debugging and repair (using explanations), where standard DL Reasoners do not provide additional help.

Finally, we are working on extending Swoop to further aid various forms of Web ontology development such as distributed, collaborative ontology evolution, the pieces of which already exist as separate Swoop components.

In this manner, with the strong platform Swoop provides for Web ontology development and its easy extensibility, Swoop is both, accessible and useful, for OWL developers and users.

5 Acknowledgments

The authors would wish to thank Ron Alford, Taowei David Wang, Vladimir Kolovski, Daniel Hewlett, and Michael Grove for their contribution in developing Swoop and the associated API's/plugins.

This work was completed with funding from Fujitsu Laboratories of America – College Park, Lockheed Martin Advanced Technology Laboratory, NTT Corp., Kevric Corp., SAIC, National Science Foundation, National Geospatial-Intelligence Agency, DARPA, US Army Research Laboratory, NIST, and other DoD sources.

6 References

1. Aditya Kalyanpur, Bijan Parsia, and James Hendler. A tool for working with web ontologies. In In Proceedings of the International Journal on Semantic Web and Information Systems, Vol. 1, No. 1, Jan - March, 2005.
2. Bijan Parsia and Evren Sirin. Pellet: An OWL DL Reasoner. Poster, In Third International Semantic Web Conference (ISWC2004), Hiroshima, Japan, November 2004.
3. Bernardo Cuenca Grau, Bijan Parsia, and Evren Sirin. Working with multiple ontologies on the semantic web. In Proceedings of the Third Internatonal Semantic Web Conference (ISWC2004). Volume 3298 Lecture Notes in Computer Science., 2004.
4. Bijan Parsia, Evren Sirin, and Aditya Kalyanpur. Debugging owl ontologies. In The 14th International World Wide Web Conference (WWW2005), Chiba, Japan, May 2005.

7 Demo Description

The live demo will consist of interaction with the Swoop[1] tool. Individuals will have the option to see either a quick full-version demo (covering both basic and advanced features mentioned in the paper) or a more in-depth demo focusing on specific advanced features such as ontology partitioning or debugging.

Sample screenshots of Swoop in action:

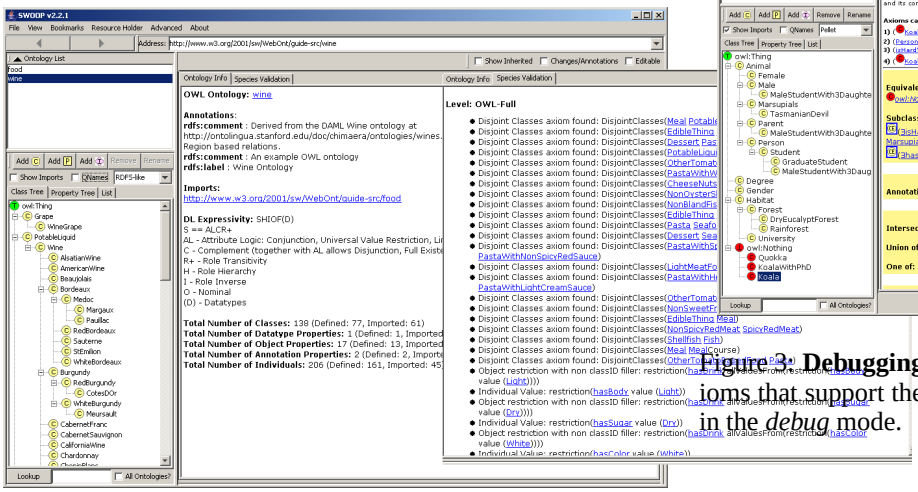

Figure 1: **Swoop Ontology Renderers**: display statistical information about the ontology, annotations, DL expressivity and OWL species level.

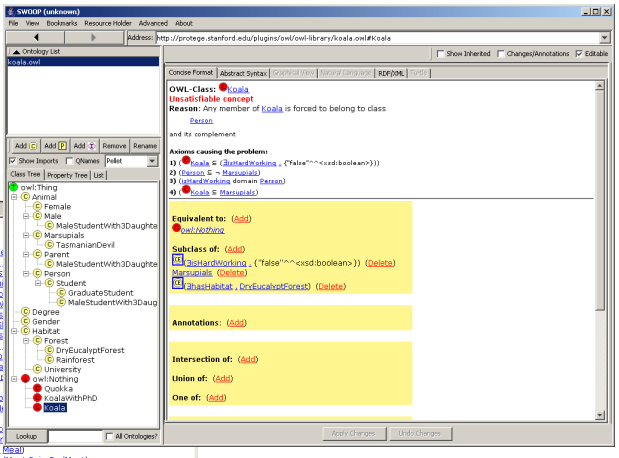

Figure 3: **Debugging Unsatisfiable Concepts:** The set of axioms that support the inconsistency of a concept is displayed in the *debug* mode.

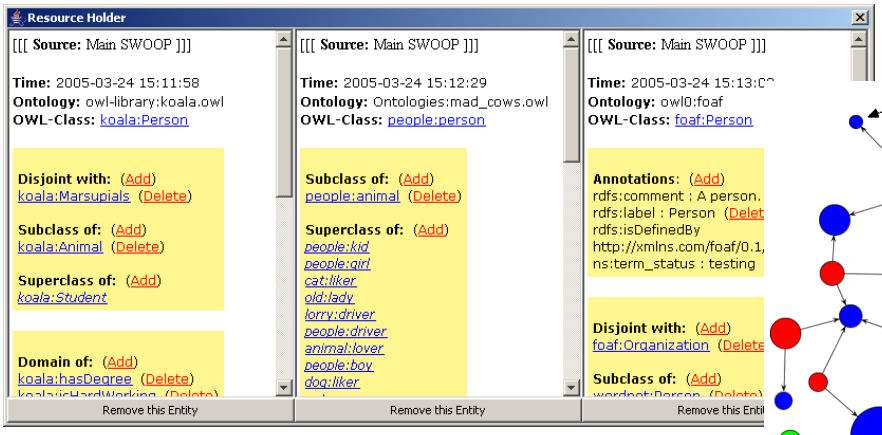

Figure 2: **Resource Holder:** used to compare 3 different definitions of the OWL Class `Person` as specified in 3 different ontologies

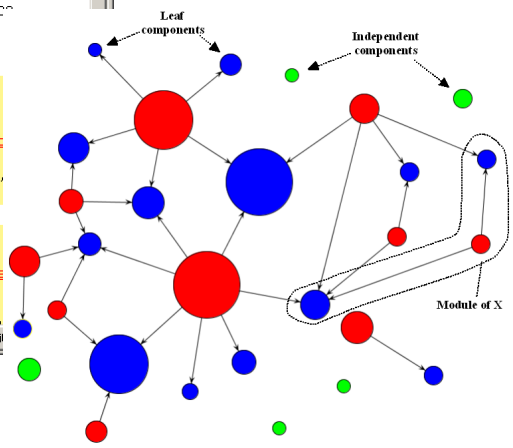

Figure 4: **Partitioning OWL Ontologies:** Visualization-plugin for inspecting modules generated after automatically partitioning OWL Ontologies into E-connections.

[1] http://www.mindswap.org/2004/SWOOP