Swoop: Design and Architecture of a Web Ontology Browser (/Editor)

(Scholarly Paper for Master's Degree in Computer Science with Non-Thesis Option, Fall 2004)

Primary Author
Aditya Kalyanpur
Dept of Computer Science
University of Maryland,
College Park 20742
aditya@cs.umd.edu

Advisor
Prof. James Hendler
Dept of Computer Science
University of Maryland,
College Park 20742
hendler@cs.umd.edu

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. Additionally, in building Swoop, we deal with several hard research problems related to ontology engineering, some of which are elaborated upon in this paper by presenting preliminary solutions (implemented in Swoop) and discussing subsequent next steps.

1. Introduction

The Web Ontology Language, OWL [WebOnt, 2004] was approved in February of 2004 as a World Wide Web Consortium (W3C) Recommendation for the publication of ontologies on the World Wide Web -- creating a standard language for the publication and exchange of ontological models on the Web. OWL reflects almost ten years of research, experimentation, and small scale deployment of Web ontologies and a number of certain features in its design were made explicitly to help realize the ideal of Web based ontologies, that is, of integrating knowledge representation with the open, global, and distributed hypermedia system of the Web, compatible with the principles of Web architecture design.

Web browsers are the ubiquitous way that people use URIs, and, even in authoring tools, the primary mental model people have of URIs is derived from their use in browsers. We take inspiration from this web-browser UI in building *Swoop*, an **ontology browser and editor**, designed specifically for use with OWL and directly supporting the use of Web-based "cultural metaphors" -- that is, based on the way people are used to interacting with documents and data in current Web applications.

2. Related Work

Most existing ontology development toolkits such as Protégé [Stanford, 2000], Oiled [Bechhofer et al, 2001], OntoEdit [Sure 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.

Alternately, we note that the entire Swoop interface and functionality could have been provided as a Website, or on top of a more full fledged Web browser such as Mozilla. There are several examples of current website-based ontology tools (e.g. Ontosaurus [Farquhar et al, 1996], WebODE [Arpírez et al, 2001]), and new ones are being developed (e.g. pOWL - http://powl.sourceforge.net). However, we have found that using a standard web-based server-client architecture for ontology engineering suffers from being slow (esp. for large ontologies, and depending on network traffic), and cumbersome for maintaining consistency while editing (e.g. trapping input errors, changing/deleting objects but reloading from browser cache etc). In addition, such tools can be difficult to extend to new functionalities via plug-in architectures (such as the one used in Swoop). In addition, most website based ontology editors use distinct HTML pages (perhaps dynamically generated) not just for each entity, but for each view of those entities. This indirection puts an uncomfortable distance between the user and the ontology itself.

3. Design Rationale & Goals

Swoop takes the standard Web browser as the UI paradigm, believing that URIs are central to the understanding and construction of Semantic Web 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 syntaxes are used to render ontologies, and an OWL reasoner 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 the Swoop user to take advantage of the Web-based features of OWL significantly more easily than the user of other ontology-editing tools.

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* [Kalyanpur et al, 2004], and expert users interested in robust ontology modeling and analysis.

4. Swoop Architecture

Swoop is based on the Model-View-Controller (MVC) paradigm. The *SwoopModel* component stores all ontology-centric information pertaining to the Swoop Workspace (currently loaded ontologies, change-logs, checkpoints) and defines key parameters used by the Swoop UI objects (such as selected OWL entity, view settings for imports, QNames etc). Additionally, a *SwoopModelListener* class is used to reflect changes in the UI based on changes in the *SwoopModel* (using a suitably defined event-notification scheme). Control is handled through a **plugin based system**, which loads new *Renderers* and *Reasoners* dynamically. The obvious advantage of a plugin framework is to ensure modularity of the code, and encourage external developers to contribute to the Swoop project easily. Finally, we note that the entire Swoop code is written in Java, maintained in a subversion repository and makes use of numerous third party libraries, the most prominent being the WonderWeb OWL API [Bechhofer et al, 2003] for parsing OWL ontologies.

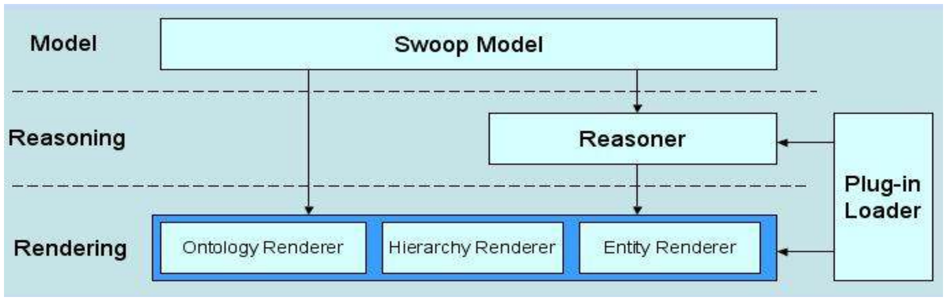
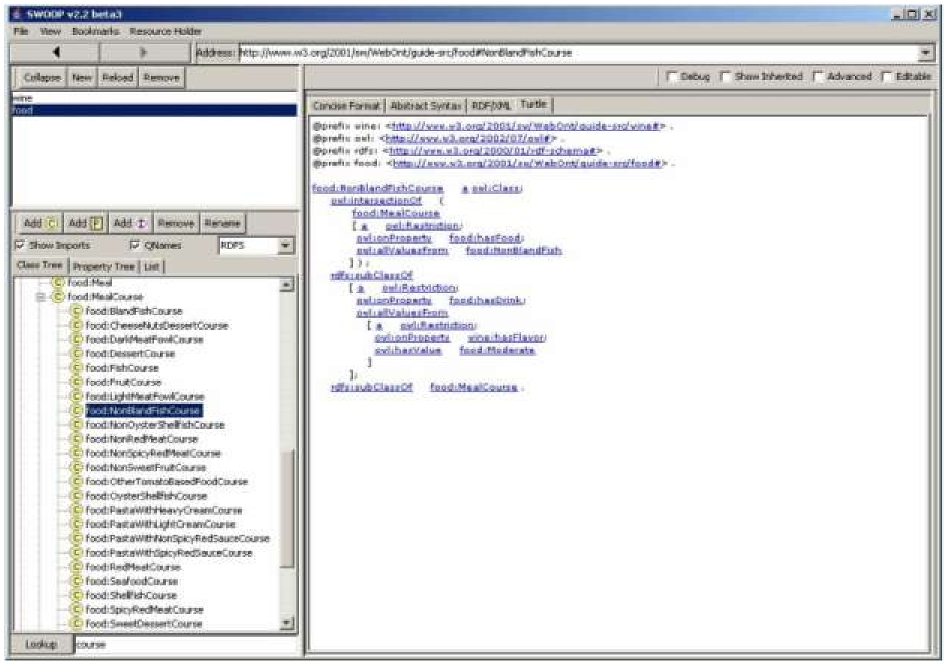


Figure 1: Swoop Architecture (Plugin-based)

5. Swoop Features

Swoop functionality is characterized by the following basic features (for an elaborate discussion of the features see [Kalyanpur et al, 2005]):


SWOOP v2.2 beta3
File View Bookmarks Resource Holder
Address: http://www.w3.org/2001/sw/WebOnt/guide-src/food#NonBlandFishCourse
Collapse New Reload Remove
wine
food
Debug Show Inherited Advanced Editable
Concise Format Abstract Syntax RDF/XML Turtle
Add C Add P Add I Remove Rename
Show Imports QNames RDFS
Class Tree Property Tree List
food:Meal
food:MealCourse
food:BlandFishCourse
food:CheeseNutsDessertCourse
food:DarkMeatFowlCourse
food:DessertCourse
food:FishCourse
food:FruitCourse
food:LightMeatFowlCourse
food:NonBlandFishCourse
food:NonOysterShellfishCourse
food:NonRedMeatCourse
food:NonSpicyRedMeatCourse
food:NonSweetFruitCourse
food:OtherTomatoBasedFoodCourse
food:OysterShellfishCourse
food:PastaWithHeavyCreamCourse
food:PastaWithLightCreamCourse
food:PastaWithNonSpicyRedSauceCourse
food:PastaWithSpicyRedSauceCourse
food:RedMeatCourse
food:SeafoodCourse
food:ShellfishCourse
food:SpicyRedMeatCourse
food:SweetDessertCourse
Lookup course

```
@prefix wine: <http://www.w3.org/2001/sw/WebOnt/guide-src/wine#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix food: <http://www.w3.org/2001/sw/WebOnt/guide-src/food#> .

food:NonBlandFishCourse    a owl:Class;
   owl:intersectionOf    (
       food:MealCourse
       [ a    owl:Restriction;
        owl:onProperty    food:hasFood;
        owl:allValuesFrom    food:NonBlandFish
      ] );
   rdfs:subClassOf
      [ a    owl:Restriction;
        owl:onProperty    food:hasDrink;
        owl:allValuesFrom
          [ a    owl:Restriction;
            owl:onProperty    wine:hasFlavor;
            owl:hasValue    food:Moderate
          ]
      ];
   rdfs:subClassOf    food:MealCourse .
```

Figure 2: Web Browser Look&Feel of Swoop

- **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. Different rendering styles, formats, and icons are used to highlight key ontological information (entity types, imported axioms, inferred axioms, changes etc) in this common pane.

- **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 given the wide range of user preferences and/or third party tool constraints.

- **Semantic Search:** Swoop takes inspiration from the hyperlink based search and cross-referencing utility present in a programming IDE such as Eclipse (*http://www.eclipse.org/*). All named entities in the code are identified and one can easily obtain (and jump directly to) useful related information such as all its references in a specific project or working set. We plan to extend this feature to support ad hoc queries (class expressions, see *concept search* in [Kalyanpur et al, 2004]).

- **Collaborative Annotation:** Swoop uses the Annotea [Kahan et al, 2001] framework as the basis of collaborative ontology development. Annotea support in Swoop is provided via a simple plug in whose implementation is based on the standard W3C Annotea protocols [Swick, 01] and 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. The default annotation types (*comment, advice, example*, etc) seem an adequate base for human oriented ontology annotations. One extension we have begun experimenting with is "*PrototypicalIllustration*", that is, a photo or drawing that represents a typical or canonical instance of the class.

- **Multimedia Markup Extension:** Swoop itself can be directly plugged in to third party semantic annotation tools such as SMORE [Kalyanpur et al, 2001], which make use of its fluid hypermedia-based UI to support rich-text, image and video markup.

6. Primary Research Focus

In this section, we list three critical ontology engineering problems that are currently being dealt with in Swoop.

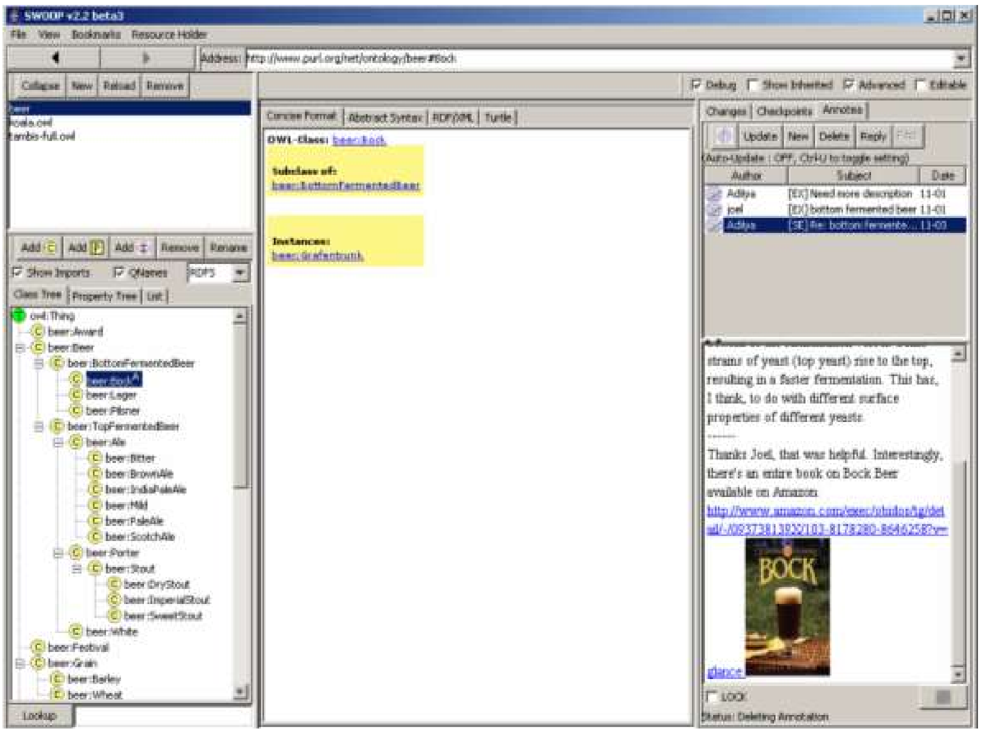
6.1 Ontology Versioning Problem: Since OWL ontologies can evolve over time, an ontology versioning mechanism is highly essential to maintain accuracy of the knowledge encapsulated by the ontology and ensure interoperability with any linked ontologies. Our ultimate goal is to build a sophisticated ontology version control system along the lines of CVS or Subversion. We have taken the first big step towards realizing this by capturing *atomic* ontology changes (i.e. at a very fine granularity level) in Swoop and providing a mechanism to annotate and exchange these changes among disparate users.

Current solution: annotated change sets

We have extended the Annotea Schema with the addition of an OWL ontology for a new class of annotations --- ontology changes (similar to [Klein et al, 2003]). The "Change" annotation defined by the Annotea projected was designed to indicate a proposed change to the annotated document, with the proposal described in HTML-marked-up natural language. In our extended ontology, change individuals correspond to specific, undoable changes made in Swoop during editing.

Swoop uses the OWL API to model ontologies and their associated entities, benefiting from its extensive and clean support for changes. The OWL API separates the representation of changes from the application of changes. Each possible change type has a corresponding Java class in the API, which are subsequently applied to the ontology (essentially, the Command design pattern). These classes allow for the rich representation changes, including metadata about the changes.

The Swoop change annotations can be published and retrieved by Annotea servers, or any other annotation distribution mechanism. The retrieved annotations can then be browsed, filtered, endorsed, recommended, and selectively accepted. It is thus possible to define "virtual versions" of an ontology, by specifying a base ontology and a set of changes to apply to it. This is a fairly new addition to Swoop, and we are just beginning to explore the implications of change tracking couple with annotations for the development of large, curated ontologies by collaborative groups of scientists or other ontology definers.


SWOOP v2.2 beta3
File View Bookmarks Resource Holder
Address: http://www.purl.org/net/ontology/beer#Bock
Collapse New Reload Remove
Debug Show Inherited Advanced Editable
Concise Format Abstract Syntax RDF/XML Turtle
Changes Checkpoints Annotea
Update New Delete Reply
Add C Add P Add I Remove Rename
Show Imports QNames RDFS
Class Tree Property Tree List
Lookup

OWL-Class: beer:Bock

Subclass of:
beer:BottomFermentedBeer

Instances:
beer:Grafentrunk

(Auto-Update : OFF, Ctrl-U to toggle setting)

Author	Subject	Date
Aditya	[EX] Need more description	11-01
joel	[EX] bottom fermented beer	11-01
Aditya	[SE] Re: bottom fermente...	11-03

strains of yeast (top yeast) rise to the top, resulting in a faster fermentation. This has, I think, to do with different surface properties of different yeasts.

Thanks Joel, that was helpful. Interestingly, there's an entire book on Bock Beer available on Amazon.
http://www.amazon.com/exec/obidos/tg/detail/-/0937381390/103-8178280-8646258?v=
glance

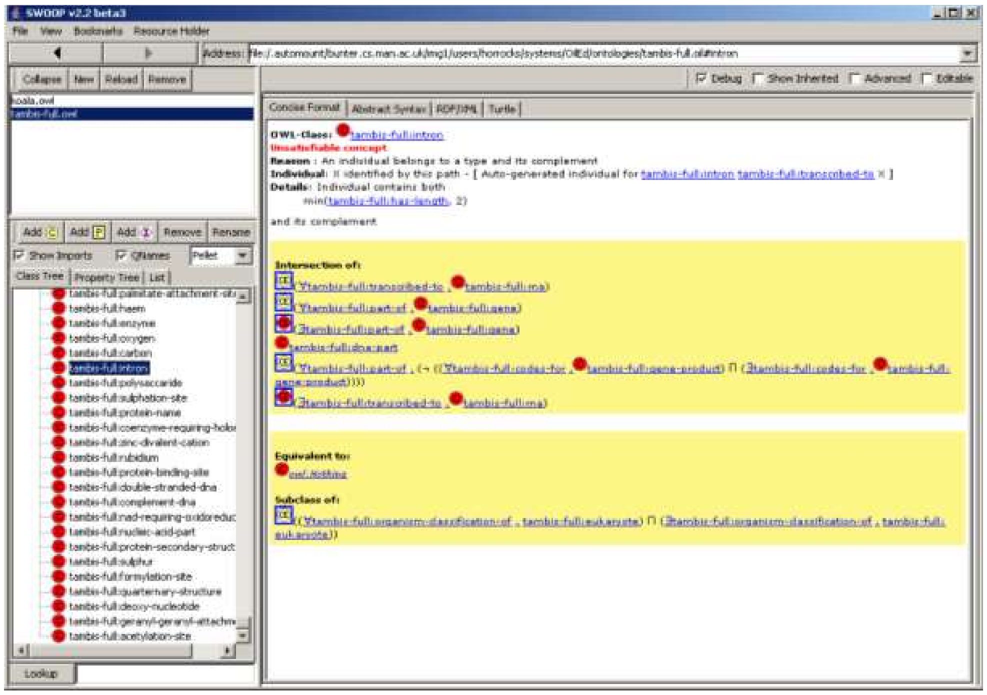
Status: Deleting Annotation

Figure 3: Create and Share Change Sets using Annotea

6.2 Ontology Debugging Problem: As an increasingly large number of OWL ontologies become available on the Semantic Web and the descriptions in the ontologies become more complicated, finding the cause of errors becomes an extremely hard task even for experts. Existing ontology development environments provide some limited support, in conjunction with a reasoner, for detecting and diagnosing errors in OWL ontologies, but typically these are restricted to the mere detection of, for example, unsatisfiable concepts.

We have integrated a number of simple debugging cues generated from our description logic reasoner, Pellet, in Swoop in order to provide a higher level of ontology debugging support.

Current solution: integration with an OWL DL reasoner
Swoop has a *debug* mode wherein the basic rendering of entities is augmented with information obtained from a reasoner (Pellet). Different rendering styles, formats, and icons are used to highlight key entities and relationships that that are likely to be helpful to debugging process. For example, all unsatisfiable named classes, and even class expressions, are marked with red icons whenever rendered --- a useful pointer for identifying dependencies between inconsistencies. Additionally, all *inferred* relationships (axioms) in a specific entity definition are *italicized* and are obviously not editable directly. Simply highlighting them separately is useful to the ontology modeler as they can (potentially) point to unintended assertions. Finally, various types of clashes are identified in the ontology and *quasi natural language* explanations (reasons) are given for clash occurrences. For more details on debugging OWL ontologies in Swoop, see [Parsia et al, 2005].



file:/.automount/bunter.cs.man.ac.uk/mg1/users/horrocks/systems/OilEd/ontologies/tambis-full.oil#intron

koala.owl
tambis-full.owl

OWL-Class: ● tambis-full:intron
Unsatisfiable concept
Reason : An individual belongs to a type and its complement
Individual: ! identified by this path - [Auto-generated individual for tambis-full:intron tambis-full:transcribed-to ✕]
Details: Individual contains both
min(tambis-full:has-length, 2)

and its complement

Intersection of:
(∀tambis-full:transcribed-to , ● tambis-full:rna)
(∀tambis-full:part-of , ● tambis-full:gene)
(∃tambis-full:part-of , ● tambis-full:gene)
● tambis-full:dna-part
(∀tambis-full:part-of , (¬ ((∀tambis-full:codes-for , ● tambis-full:gene-product) ⊓ (∃tambis-full:codes-for , ● tambis-full:gene-product))))
(∃tambis-full:transcribed-to , ● tambis-full:rna)

Equivalent to:
● owl:Nothing

Subclass of:
((∀tambis-full:organism-classification-of , tambis-full:eukaryote) ⊓ (∃tambis-full:organism-classification-of , tambis-full:eukaryote))

● tambis-full:palmitate-attachment-sit
● tambis-full:haem
● tambis-full:enzyme
● tambis-full:oxygen
● tambis-full:carbon
● tambis-full:intron
● tambis-full:polysaccaride
● tambis-full:sulphation-site
● tambis-full:protein-name
● tambis-full:coenzyme-requiring-holo
● tambis-full:zinc-divalent-cation
● tambis-full:rubidium
● tambis-full:protein-binding-site
● tambis-full:double-stranded-dna
● tambis-full:complement-dna
● tambis-full:nad-requiring-oxidoreduc
● tambis-full:nucleic-acid-part
● tambis-full:protein-secondary-struct
● tambis-full:sulphur
● tambis-full:formylation-site
● tambis-full:quarternary-structure
● tambis-full:deoxy-nucleotide
● tambis-full:geranyl-geranyl-attachm
● tambis-full:acetylation-site

Figure 4: Ontology *Debug* Mode in Swoop (uses Pellet)

6.3 Ontology Refactoring Problem
The problem of refactoring large, complex ontologies into smaller, more manageable units is a hard research problem. In [Cuenca-Grau et al, 2004], the authors outline a semantically robust methodology for refactoring and mapping ontologies (overcoming the limitations of *owl:imports*), known as *e-connections*. Swoop has preliminary support to parse and display e-connected ontologies independently. As a next step, we plan to investigate techniques to suggest refactoring options to ontology modelers, and accordingly provide wizards to semi-automate the refactoring process.

7. 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. Moreover, a formal evaluation of the features needs to be done by performing usability studies and comparing it against existing ontology engineering tools.

8. Acknowledgments

The authors wish to thank *Bijan Parsia* and *Evren Sirin* for their invaluable contribution in the design and implementation of Swoop.

9. References

[Arpirez et al, 2001] Arpirez, J., Corcho, O., Fernandez-Lopez, M., Gomez-Perez, A. *WebODE: A Scalable Ontological Engineering Workbench*. First International Conference on Knowledge Capture (K-CAP) (2001)

[Bechhofer et al, 2001] Bechhofer, S., Horrocks, I., Goble, C., Stevens, R.: *OilEd: a reasonable ontology editor for the Semantic Web*. Proceedings of KI2001, Joint German/Austrian conference on Artificial Intelligence (2001)

[Bechhofer et al, 2003] Bechhofer, S., Lord, P., Volz, R. *Cooking the Semantic Web with the OWL API*. Proceedings of the International Semantic Web Conference (2003)

[Cuenca-Grau et al, 2004] Cuenca-Grau, B., Parsia, B., Sirin, E.: *Working with Multiple Ontologies on the Semantic Web*. Proceedings of the Third International Semantic Web Conference (ISWC) (2004)

[Farquhar et al, 1996] Farquhar, A., Fickas, R., Rice, J. *The Ontolingua server: A Tool for Collaborative Ontology Construction*. Proceedings of the 10th Ban# Knowledge Acquisition for Knowledge Based System Workshop (KAW95) (1996)

[Kahan et al, 2001] Kahan, J., Koivunen, M.R., Prud'Hommeaux, E., Swick, R.: Annotea: *An Open RDF Infrastructure for Shared Web Annotations*. Proc. of the WWW10 International Conference (2001)

[Kalyanpur et al, 2001] Aditya Kalyanpur, Bijan Parsia, James Hendler and Jennifer Golbeck, "*SMORE - Semantic Markup, Ontology and RDF Editor*". Technical Report http://www.mindswap.org/~aditkal/SMORE.pdf

[Kalyanpur et al, 2004] Aditya Kalyanpur, Nada Hashmi, Jennifer Golbeck, Bijan Parsia *Lifecycle of a Casual Web Ontology Development Process*. Proceedings of the WWW2004 Workshop on Application Design, Development and Implementation Issues in the Semantic Web, May 18, 2004 , NYC, USA.

[Kalyanpur et al, 2005] Aditya Kalyanpur, Bijan Parsia, James Hendler. *A Tool for*

Working with Web Ontologies, To Appear in the International Journal on Semantic Web and Information Systems, 2005

[Klein et al, 2003] Klein, M., Noy, N.: *A Component-based Framework for Ontology Evolution*. Workshop on Ontologies and Distributed Systems at IJCAI (2003)

[Parsia et al, 2005] Bijan Parsia, Evren Sirin, Aditya Kalyanpur. *Debugging OWL Ontologies*, Submitted to WWW 2005

[Stanford, 2000] Noy, N., Sintek, M., Decker, S., Crubezy, M., Fergerson, R., Musen, M. *Creating semantic web contents with Protege-2000*. IEEE Intelligent Systems (2001)

[Sure et al, 2002] Sure, Y., Erdmann, M., Angele, J., Sta#, S., Studer, R., Wenke, D.: *OntoEdit: Collaborative ontology development for the Semantic Web*. Proceedings of the International Semantic Web Conference (ISWC) (2002)

[WebOnt, 2004] Dean, M. and Schreiber, G. *OWL Web Ontology Language Reference W3C Recommendation.* http://www.w3.org/TR/owl-ref/, February, 2004.