Integrating Deep and Shallow Semantic Structures in Open Ontology Forge

Nigel Collier, Ai Kawazoe, Asanobu Kitamoto,
Tuangthong Wattarujeekrit, Yoko Mizuta, Anthony Mullen
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, 101-8430 Japan
E-mail:{collier,zoeai,kitamoto,tuang,ymizuta,mullen}@nii.ac.jp

Abstract

The Semantic Web initiative has enabled a common infrastructure in knowledge representation for sharing knowledge of ontologies and instances through languages such as Resource Description Framework (RDF) and the Web Ontology Language (OWL). In this paper we present a framework for combining the shallow levels of semantic description commonly used in information extraction with the deeper semantic structures available in such ontology representation languages. In this approach named entities are considered as instances of concepts in a defined taxonomy, coreference relations as identity pools containing named entities and coreferring expressions, and events as predicate-argument frames. The framework is being implemented within the Open Ontology Forge software for annotating media, currently text and images, in Web pages and locally stored document collections. We discuss the knowledge framework, some features of the system, and a road map for future development.

1 Introduction

Annotation of resources is a major requirement of many academic and industrial projects but so far few tools have been freely available which combine creation of ontologies with the ability to populate them using annotations linked to texts and images in Web-based documents. In this paper we outline Open Ontology Forge (OOF) , an ontology editor and multi-modal content annotation tool. We discuss the knowledge framework, the process model, some features of the system, and summarize our experience so far through case studies on annotating scientific texts in the molecular biology domain and initial experiences of annotating images in the cultural heritage domain.

Ontologies are rapidly emerging as 'engineering artifacts' with standardized languages such as Resource Description Framework (RDF) (Lassila and Swick, 1999) and RDF Schema (Brickley and Guha, 2000) that have been developed by the Semantic Web initiative (Berners-Lee et al., 1999) led by the World Wide Web Consortium (W3C). While ontologies allow for improved knowledge sharing through a common vocabulary and query language, the acquisition and maintenance of that vocabulary is a key issue due to the size and complexity of many ontologies as well as the profile of the users who do this task.

There are two key issues to consider here: the first is the profile of users who perform the annotations and the second is how to reduce the effort in making annotations. With regard to the user profile we should consider that it is desirable that semantic annotation should become a common activity with tools that are as intuitive and easy to use as those which we now have on our desktop such as Web design, image editing, and word processing software. To accomplish this we have to emphasize semantic markup functions for ordinary users who are experts in their domains but not necessarily familiar with the concepts and techniques of knowledge engineering. What we are aiming to achieve here is to elicit a domain description from the user within the environment of a familiar user interface while embedding issues such as knowledge acquisition methodology and formal representation seamlessly within the software process.

With regard to the need to reduce effort in annotation we consider that information extraction (IE) has an important part to play. IE systems are now well developed for capturing low level semantics inside texts such as named entities and coreference (identity) expressions. IE however does not offer a sufficiently formal framework for specifying relations between concepts, assuming for example that named entities are instances of a disjoint set of concepts with no explicit relations. Without such formalization it is difficult to consider the implications of concept class relations when applying named entity outside the news domain. To look at this another way, by making the deep semantics explicit we offer a way to sufficiently abstract IE to make it portable between domains.

As a motivating example we consider information access to research results contained in scientific journals and papers. The recent proliferation of scientific results available on the Web means that scientists and other experts now have far greater access than ever before to the latest experimental results. While a few of these results are collected and organized in databases most remain in free text form. Efficient tools such as the commercial search engines have been available for some years to help in the document location task but we still do not have effective tools to help in locating facts inside documents. Such tools require a level of understanding far beyond simple keyword spotting and for this reason have taken longer to develop. *Access* to information is therefore now limited by the time the expert spends reading the whole text to find the key result before it can be judged for relevance and synthesized into the expert's understanding of his/her *domain* of knowledge. By explicitly describing the concepts and relations inside the scientific domain and then annotating a few examples of texts we expect that machine learning will enable new instances to be found in unseen documents, allowing computer software to have an intelligent understanding of the contents, thereby aiding the expert in locating the information he/she requires.

The focus of this paper is on three topics: firstly to summarize the OOF system which allows ontologies and annotations to be created cooperatively within a domain community; secondly to outline a metadata scheme for annotations that can provide linkage between the ontology and a text; and thirdly to present a discussion of the process and user interface aspects of the software. We motivate our discussion with examples taken from the domain of functional molecular biology.

2 The ontology forge system

In this section we will outline a few details about OOF including the meta-data scheme and process model. The knowledge model is deliberately rather conventional for purposes of compatibility and has several similarities to other ontology editors such as Protege-2000 (Noy et al., 2001) and OntoEdit (Sure et al., 2002).

An OOF ontology is centred around a frame-based knowledge model consisting of classes, properties (slots) and annotations. Classes are related through subsumption in a simple taxonomy. Properties are a binary relation between a domain (a class) and a range (a value data type). While properties are defined globally within OOF as part of the conceptualization process they are not regarded as part of the global name space for purposes of exporting the knowledge base. In other words they are not referrable independently of the class to which they are attached.

What distinguishes OOF from many other ontology editors is its ability to provide support for content annotation. An annotation in OOF is regarding as *instance + linkage + tracking*, with linkage and tracking information automatically calculated and maintained by OOF. An annotation is regarded as belonging to a single specified class and cannot have its type changed directly (this must be done by redefining the parent class). Annotations can be grouped within what we call 'coreference pools' that define identity relations and refer to the same concept.

It should be noted that instances in OOF ontologies are not an abstract concept, but a surface-level representation of the concept appearing in the document, in the form of texts or images. Eventually we hope to handle various other modes of content. Links relate document annotations with classes in ontology, and tracking information relates annotations to source documents. Links and tracking information are providers of context in which the instance appears, and helps IE tasks that use machine learning from contexts, and enables users to judge the reliability of the annotation for the instance by providing the information about who made which annotation on which document, when the annotation made, and if the author is sure about the annotation.

2.1 Annotation meta-data

In this section we will focus on the specific characteristics of our annotation scheme.

The root class in an Ontology Forge ontology is the annotation. The user does not need to be explicitly aware of this and all classes that are defined in OOF will inherit the annotation class properties from the parent so that when instances are declared as annotations in a base

Web document, the instance will have many of the property values entered automatically by OOF such as linkage information. Basically the user is free to create any classes that help define knowledge in their domain and to let OOF handle the maintenance of linkage information.

We briefly now describe the root annotation class:

context This relates an Annotation to the resource to which the Annotation applies and takes on an XPointer value.

ontology_id Relates an Annotation to the ontology and class to which the annotation applies.

conventional_form The conventional form of the annotation (if applicable) as described in the PIA Annotation Guidelines.

identity_id Used for creating coreference chains between annotations where the annotations have identity of reference. From a linguistic perspective this basically assumes a symmetric view of coreference where all coreference occurrences are considered to be equal.

constituents This is used to capture constituency relations between annotations if required.

orphan This property takes only Boolean values corresponding to 'yes' and 'no'. After the annotation is created, if it is later detected that the annotation can no longer be linked to its correct position in doc_id, then this value will be set to 'yes' indicating that the linkage (in context) needs correcting.

author The name of the person, software or organization most responsible for creating the Annotation. It is defined by the Dublin Core (dub, 1999) 'creator' element.

created The date and time on which the Annotation was created. It is defined by the Dublin Core 'date' element.

modified The date and time on which the Annotation was modified. It is defined by the Dublin Core 'date' element.

sure This property takes only Boolean values corresponding to 'yes I am sure' and 'no I am not sure' about the assignment of this annotation. Used primarily in post-annotation processing.

comment A comment that the annotator wishes to add to this annotation, possibly used to explain an unsure annotation. It is defined by the Dublin Core 'description' element.

Currently the two representation formats supported by OOF are RDF and in-line XML for annotations only. The later is useful for training IE applications using supervised machine learning that require labelled examples but do not require the schema. In the future we plan on extending this with OWL (Web Ontology Language [1]).

When exporting to RDF there are some considerations with regard to data types. Due to the lack of defined data types in RDF we make use of several data types in the Dublin Core name space for defining annotation property values. In OOF we also allow users to use a rich set of data type values using both Dublin Core and also XML Schema name space for integers, floats etc. Users can also define their own enumerated types using OOF.

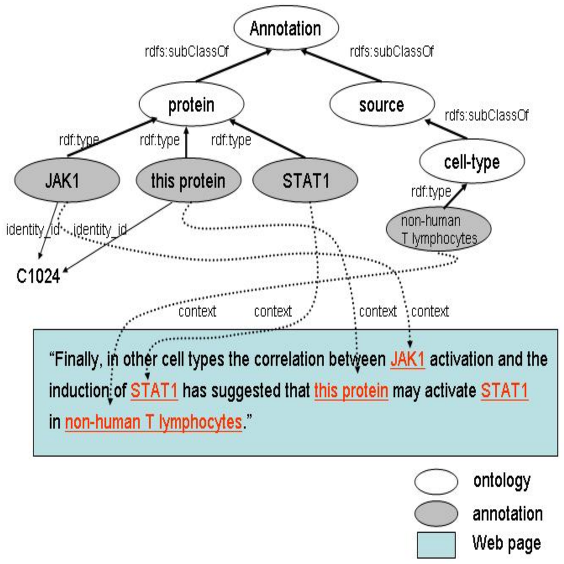
A selected view of an annotation can be seen in Figure 1 showing linkage into a Web document and part of the ontology hierarchy. One point to note is the implementation of coreference relations between *JAK1* and *this protein.* This helps to maximize information about this object in the text.

2.2 Process model

The basic process model is outlined in Table 1 extrapolated from standard ontology engineering processes (e.g. see (Gómez-Pérez, 1998)). This is informally summarized below

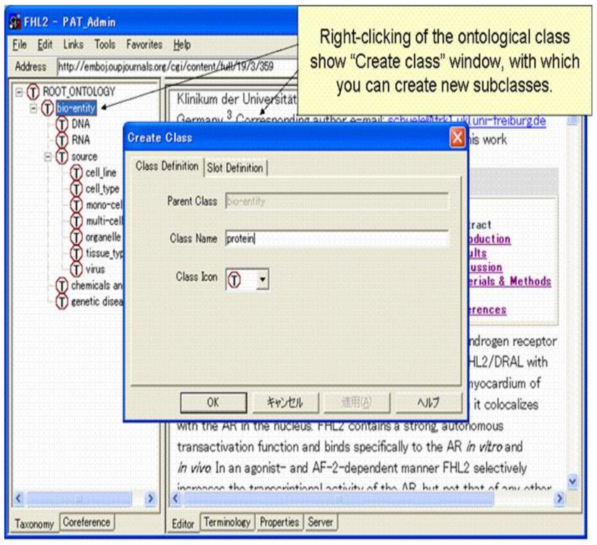
- *planning* - decide how to allocate project resources, what tasks should be performed, setting quality criteria
- *knowledge acquisition* - collect useful sources of information that will be used in the building stage
- *building* - structure domain knowledge, merge existing sources of domain knowledge, represent the domain knowledge, document the knowledge and the construction process
- *evaluation* - use the ontology and test it according to the quality criteria

[1]http://www.w3.org/TR/owl-features/



"Finally, in other cell types the correlation between JAK1 activation and the induction of STAT1 has suggested that this protein may activate STAT1 in non-human T lymphocytes."

Figure 1: Overview of an annotation.



Right-clicking of the ontological class show "Create class" window, with which you can create new subclasses.

Parent Class: bio-entity

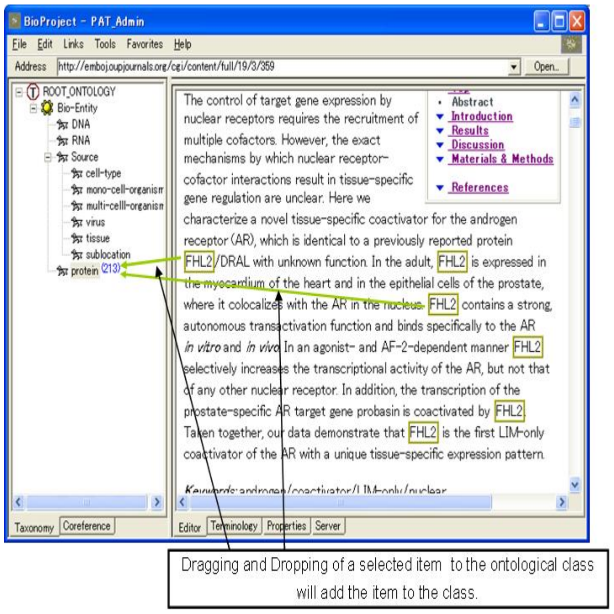
Class Name: protein

Figure 2: Basic view of OOF showing the conceptualization stage with taxonomy in the right window, a text to be annotated in the left window.

It should be noted of course that this is not a purely linear procedure but occurs within an ontology life cycle. In particular the user actions involved in building the ontology will have their own cycles. For example training and test of the IE system may involve re-annotation by the user in order to improve the performance of the IE system.

3 User interface and example

As shown in the screenshot of Figure 2, the user in the right hand window has a full Web-browser view of the Web page including images, and in the left hand window can view the ontology (and coreference pools under ontological classes when the coreference annotation mode is selected). Class-subclass relations are visualized in a tree in OOF. Coreference pools are visualized as terminal nodes of the ontological tree in OOF interface. Users can create the ontology in the fashion of extending the root class, defining new class names under it.



The control of target gene expression by nuclear receptors requires the recruitment of multiple cofactors. However, the exact mechanisms by which nuclear receptor–cofactor interactions result in tissue-specific gene regulation are unclear. Here we characterize a novel tissue-specific coactivator for the androgen receptor (AR), which is identical to a previously reported protein FHL2/DRAL with unknown function. In the adult, FHL2 is expressed in the myocardium of the heart and in the epithelial cells of the prostate, where it colocalizes with the AR in the nucleus. FHL2 contains a strong, autonomous transactivation function and binds specifically to the AR *in vitro* and *in vivo*. In an agonist- and AF-2-dependent manner FHL2 selectively increases the transcriptional activity of the AR, but not that of any other nuclear receptor. In addition, the transcription of the prostate-specific AR target gene probasin is coactivated by FHL2. Taken together, our data demonstrate that FHL2 is the first LIM-only coactivator of the AR with a unique tissue-specific expression pattern.

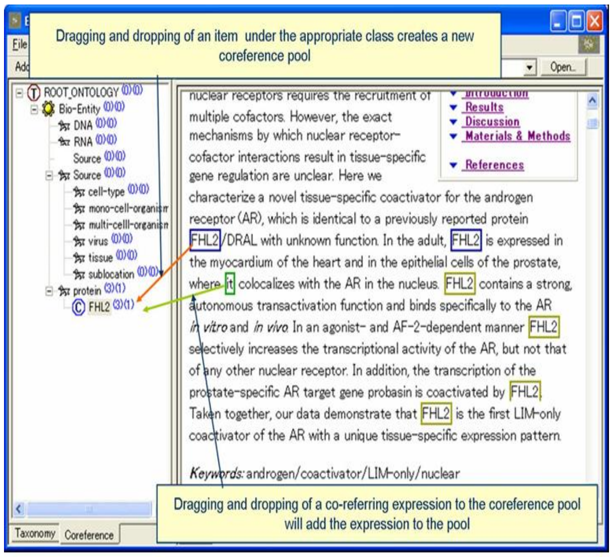
Dragging and Dropping of a selected item to the ontological class will add the item to the class.

Figure 3: Named entity (terminology) annotation.

For text annotation, the software has two modes: the named entity annotation mode and the coreference annotation mode. As shown in Figure 3, when the named entity annotation mode is selected, users can add the selected part of the text in the right pane to an ontological class by dragging and dropping the text to the class. The texts annotated in this way are highlighted with yellow box. This procedure is semi-automatic: when a user captures an instance, OOF automatically searches for similar instances and offers the user the chance to annotate these too, saving the user a lot of time in duplicated work. Property values related to linkage and tracking are automatically adjusted for each new instance.

#	Development stage	Substage	User action in OOF
1	Planning		
2	Knowledge acquisition		browse Web pages
3	Building	Conceptualization	create taxonomy
		Populate and link	annotate terminology & images
			define coreference pools
			train IE*
			test IE*
		Represent	export
4	Evaluate		

Table 1: Basic process model in constructing and populating OOF ontologies. * indicates that this task is now being implemented.



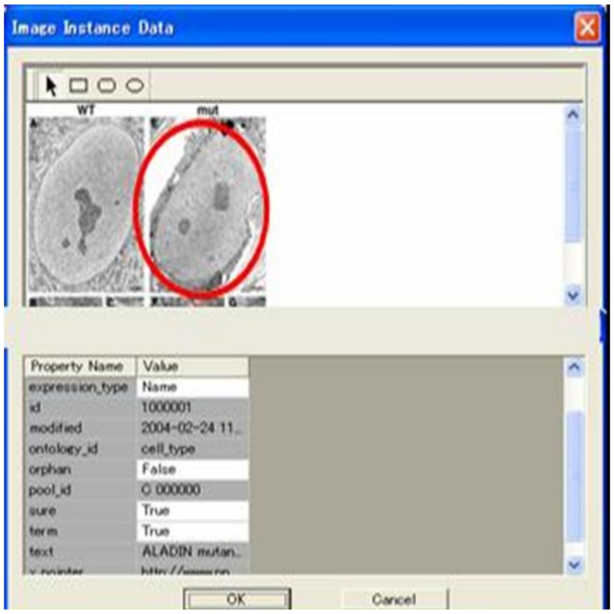
Dragging and dropping of an item under the appropriate class creates a new coreference pool

nuclear receptors requires the recruitment of multiple cofactors. However, the exact mechanisms by which nuclear receptor-cofactor interactions result in tissue-specific gene regulation are unclear. Here we characterize a novel tissue-specific coactivator for the androgen receptor (AR), which is identical to a previously reported protein FHL2/DRAL with unknown function. In the adult, FHL2 is expressed in the myocardium of the heart and in the epithelial cells of the prostate, where it colocalizes with the AR in the nucleus. FHL2 contains a strong autonomous transactivation function and binds specifically to the AR *in vitro* and *in vivo*. In an agonist- and AF-2-dependent manner FHL2 selectively increases the transcriptional activity of the AR, but not that of any other nuclear receptor. In addition, the transcription of the prostate-specific AR target gene probasin is coactivated by FHL2. Taken together, our data demonstrate that FHL2 is the first LIM-only coactivator of the AR with a unique tissue-specific expression pattern.

Keywords: androgen/coactivator/LIM-only/nuclear

Dragging and dropping of a co-referring expression to the coreference pool will add the expression to the pool

Figure 4: Creation of a coreference pool.



Property Name	Value
expression_type	Name
id	1000001
modified	2004-02-24 11...
ontology_id	cell_type
orphan	False
pool_id	C 000000
sure	True
term	True
text	ALADIN mutan...
x_pointer	http://www

Figure 5: The image region specification window.

In coreference annotation mode shown in Figure 4 annotators can create coreference pools, sets of co-referential expressions, by drag-and-dropping their members from the text window. OOF checks consistency of properties among the members of coreference pool.

For image annotation, OOF has a window for selecting a part of image and editing properties as in Figure 5. Users can select a part of image by clipping it in basic shapes available in Scalable Vector Graphics (SVG).

The selected part of image can be linked to an ontological class or a coreference pool in the same manner as texts as shown in Figure 6.

4 Related work

There were several starting points for our work. Perhaps the most influential was the Annotea project (Kahan et al., 2000) which provided an annotation hosting service based on RDF Schema. Annotations in Annotea can be considered as a kind of electronic 'Post-it' for Web pages, i.e. a remark that relates to a URI. Annotea has several features which our design has used including: (a) use of XLink/XPointer linkage from annotations to the place in the document where a text occurs; (b) use of generic RDF schema as the knowledge model enhanced with types from Dublin Core.

However, there are several major differences between our work and Annotea including our

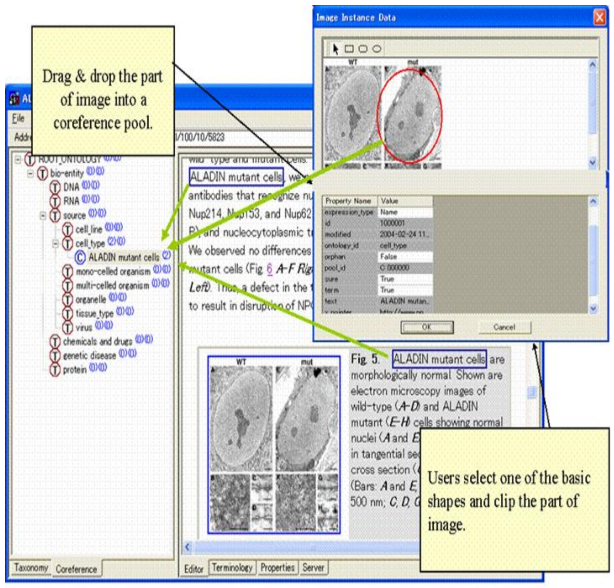

Drag & drop the part of image into a coreference pool.
Users select one of the basic shapes and clip the part of image.
Image Instance Data
ALADIN mutant cells
Fig. 5. ALADIN mutant cells are morphologically normal.
OK
Cancel
Taxonomy
Coreference
Editor
Terminology
Properties
Server

Figure 6: Annotating an image region and its coreference with text through the coreference pool.

focus on aiding annotation through the use of information extraction, various levels of support for cooperative development of annotations and ontologies, the use of domain groups and explicit roles and access rights for members.

Also influential in our development has been Protege-2000 (Noy et al., 2001) and other ontology editors such as Ont-O-Mat (Handschuh et al., 2001) which provide many of the same features as OOF including ontology editing. While Protege-2000 provides a rich environment for creating ontologies it provides limited support for large-scale instance capturing and collaborative development. Ont-O-Mat and OntoEdit from the University of Karlsruhe provide similar functionality to OOF but does not link instances back to the text or have the built-in design focus on collaborative development and information extraction.

5 Conclusion and future plans

In this paper we have presented an annotation scheme that links concepts in ontologies with instances in texts using a system that conforms to RDFS. There are many benefits of linking IE to deep semantic descriptions including abstract of knowledge and portability for IE, and automated instance capturing for Semantic Web. OOF is available freely for download[2].

OOF has been in use for over a year as part of our annotation of scientific text collections in the molecular biology domain capturing named entities and coreference expressions. It is now also being user tested for annotating images in the cultural heritage domain.

Future work will focus on integrating IE into the system through development of a standard API so that other groups can plug-in their automatic annotation models, enabling multi-document annotation, improved methods for visualizing and searching annotations and enhancing the terminology processing support.

Acknowledgements

We are very grateful to several of our colleagues for useful discussions and feedback related to some of the ideas contained in this paper. This includes Asao Fujiyama (NII), Shoko Kawamoto (NAIST), Koichi Takeuchi (Okayama University), and Norihiro Ogata (Osaka University). Development of Open Ontology Forge has been made possible by various grants generously given by the National Institute of Informatics.

References

T. Berners-Lee, M. Fischetti, and M. Dertouzos. 1999. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web.* Harper, San Francisco, September. ISBN: 0062515861.

D. Brickley and R. V. Guha. 2000. Resource Description Framework (RDF) schema specification 1.0, W3C candidate recommendation. http://www.w3.org/TR/2000/CR-rdf-schema-20000327, 27th March.

1999. Dublin core metadata element set, version 1.1: Reference description. Technical Report, Dublin Core Metadata Initiative, http://purl.org/DC/documents/rec-dces-19990702.htm, 1999.

A. Gómez-Pérez. 1998. Knowledge sharing and reuse. In J. Liebowitz, editor, *Handbook of Applied Expert Systems.* CRC Press.

S. Handschuh, S. Staab, and A. Maedche. 2001. CREAM - creating relational metadata with a component-based, ontology-driven annotation framework. In *First International Conference on Knowledge Capture (K-CAP 2001), Victoria, B.C., Canada*, 21 – 23 October.

[2]http://research.nii.ac.jp/~collier/resources/OOF/OOF.html

J. Kahan, M.R. Koivunen, E. Prud'Hommeaux, and R.R. Swick. 2000. Annotea: An open RDF infrastructure for shared web annotations. In *the Tenth International World Wide Web Conference (WWW10)*, pages 623–630, May 1 – 5.

A. Lassila and R. Swick. 1999. Resource Description Framework (RDF) Model and Syntax Specification, World Wide Web Consortium Recommendation. http://www.w3.org/xml/TR/REC-rdf-syntax, 22nd February.

N. F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson, and M. A. Musen. 2001. Creating semantic web contents with Protégé-2000. *IEEE Intelligent Systems*, 16(2):60–71.

Y. Sure, M. Erdmann, J. Angele, S. Staab, S. Studer, and D. Wenke. 2002. Ontoedit: Collaborative ontology engineering for the semantic web. In I Horrocks and J. Hendler, editors, *The Semantic Web - ISWC 2002: First International Semantic Web Conference, Sardinia, Italy*, Lecture notes in Computer Science. Springer-Verlag, June 9–12. ISBN 0302-9743.