22-8-2005

Peer-to-peer based ontology editing

P. Becker
University of Wollongong, uow@becker.edu.au

P. Eklund
University of Wollongong, peklund@uow.edu.au

N. Roberts
University of Wollongong

Peer-to-peer based ontology editing

Abstract

The paper develops software to exploit a protocol for collaborative ontology editing based on RDF and using a Peer-to-Peer (P2P) networking architecture. The protocol implements a voting mechanism embedded into the RDF data itself, using a mixed initiative design for notification. This is implemented as extensions to an ontology browser called ONTORAMA1. The P2P approach is compared to the classic ontology editing approaches and the special requirements of the ontology editing environment are discussed.

Publication Details

Becker, P., Eklund, P. W. & Roberts, N. (2005). Peer-to-peer based ontology editing. Proceedings of the International Conference on Next Generation Web Services Practices (NWeSP 2005) (pp. 259-264). Los Alamitos, CA: IEEE. Copyright IEEE 2005.

Peer-to-Peer Based Ontology Editing

Peter Becker, Peter Eklund and Natalyia Roberts
School of Economics and Information Systems
The University of Wollongong
Australia, peklund@uow.edu.au

Abstract

The paper develops software to exploit a protocol for collaborative ontology editing based on RDF and using a Peer-to-Peer (P2P) networking architecture. The protocol implements a voting mechanism embedded into the RDF data itself, using a mixed initiative design for notification. This is implemented as extensions to an ontology browser called ONTORAMA[1]. *The P2P approach is compared to the classic ontology editing approaches and the special requirements of the ontology editing environment are discussed.*

1. Introduction

The aim of one large on-line ontology of semantic data is unachievable so smaller (often domain specific) Web-based knowledge bases have emerged [3]. As a result, distributing metadata descriptions identifying resources and their properties is problematic. Different systems use different terminologies to describe the same object or idea. Alternatively, a single descriptor can be associated with a number of different objects or ideas. Many ontologies also allow the definition of inference rules, conceptualizing the data encountered into a domain context and in so doing relate it to the task at hand. Ontologies have other uses: (i) as a common vocabulary for people working in the same domain; (ii) as a tool to enhance document search via query expansion or as a way of determining the "aboutness" of a document through the analysis of the lexical terms.

There are several servers available to experiment with ontology content. Ontology servers are often also knowledge base servers but the emphasis on the ontology highlights the fact that they permit Web users to modify the ontology part of the knowledge base which other knowledge base servers may not allow. The ONTOLINGUA server[2] was the first ontology server and remains active as the HPKB[3]. ONTOLINGUA permits the re-use of KIF (Knowledge Interchange Format) files. ONTOSAURUS[4] is another Web-accessible ontology server which permits each user to build or edit ontological content. TADZEBAO and WEBONTO[5] support some synchronous cooperation between co-temporal users (they can exchange multimedia messages and be warned of each other's actions). In WEBKB-2[6] knowledge from the various users is not stored into loosely connected ontologies but integrated into a single knowledge base with WORDNET as its backbone. In that system, the cooperative building of the knowledge base is supported via the enforcement of editing rules.

Each ontology server above adopts a client/server approach. This exhibits a number of problems: (i) it can induce performance bottlenecks when many clients are connected; (ii) it needs constant maintenance via authentication; (iii) it must be robust and secure from compromise; (iv) collaboration is not easily facilitated – users working on the same part of an ontology may not be aware of each others work unless a subscription system is in place; (v) different ontology servers use different formats to store their ontologies, this impedes system interoperability and knowledge sharing; (vi) there is the scalability and reliability problem, as systems increase in size so too their maintenance. On the other hand, since knowledge is de-centralized, there is a strong argument that so too must the Semantic Web be de-centralized – at least there is no momentum for centralized metadata management so a more distributed approach is likely to be preferred in many knowledge communities.

The paper is structured as follows. First, we elaborate on ontology editing, what research in this area has achieved and how our design is adapted from previous work in distributed collaboration. Secondly, we present an overview of P2P systems. Many of the features for sharing ontological content are gleaned from other successful P2P file sharing systems. We characterize our work in the context of the feature space of well-known P2P systems with the additional

1 *http://www.ontorama.com*
2 *http://www.ksl.stanford.edu/software/ontolingua/*
3 *http://lalab.gmu.edu/Projects/HPKB/HPKB-index.htm*
4 *http://www.isi.edu/isd/ontosaurus.html*
5 *http://kmi.open.ac.uk/projects/webonto*
6 *http://www.webkb.org*

Proceedings of the International Conference on Next Generation Web Services Practices (NWeSP'05)
0-7695-2452-4/05 $20.00 © 2005 IEEE

(and special) requirement of ontology editing. Thirdly, the high-level mechanisms for collaborative P2P ontology editing are developed. The idea of voting via assertions and rejections on content gives a mechanism to control ontology update and a principled approach to knowledge-based version control management. Since our solution is embedded within the ONTORAMA ontology browser [5], the fourth section of this paper describes ONTORAMA. We spent time in the fifth section describing the implementation details, demonstrating the operation of the P2P editor in use cases.

2. Ontology Editing

Maintaining, searching and navigating ontologies has a long research history. Dominigue [4] created a tool called WEBONTO for developing and maintaining ontologies. It includes functions such as visualization, browsing and editing. The models that are used for describing ontologies are OCML. Other projects have implemented different ontology servers and tools for editing, either as Web-based frontends, accessed by a browser, or as GUI applications. These include PROTÉGÉ [10], ONTOEDIT [11] and OILED [2]. The ONTOLINGUA server also has a number of features supporting collaboration [6]. It supports multiple ontologies with a search functionality, informal descriptions as an annotation tool and notifications about changes.

Mintra [9] present a toolkit called ONION to help domain experts bridge the gap between smaller domain ontologies. Prior to ONION, most research on ontology construction focused on tools for building a single global ontology which was not manageable according to most authors. Arumugam [1] is the first attempt to use a totally distributed environment to work with ontologies and the authors present their work with the P2P Semantic Web (PSW). This is an extension to the application INFOQUILT[7] allowing users to create, maintain, and control sharing of ontologies in a P2P environment. Although it allows users to make additions to ontologies, the focus is maintaining, sharing and retrieving other ontologies. The ontologies are edited via textual input. In a collaborative environment different users work on different ontologies and it is important that there is a way of reusing ontologies. This is called *ontology integration* and can be accomplished either by *merging* or *term alignment*. It is important to distinguish these two approaches. Merging means that one new ontology is created from n existing ontologies, while alignment creates links between ontologies so that they can be used as one. In this paper, we are focused on the problem of ontological term alignment, treating the ontology on the local client as separate entity, aligned to the others by common URIs.

7 *http://lsdis.cs.uga.edu/proj/iq/iq.html*

3. P2P Systems

In the last couple of years the processing and networking capacity of client computers has increased to a point where it is feasible to run network services on any machine and create systems to share files within a group of peers: called *peer-to-peer* or P2P systems. The P2P Working Group defines peer-to-peer as "sharing of computer resources & services by direct exchange between systems. These resources and services include the exchange of information, processing cycles & disk storage for files" [12]. Popular P2P systems, like NAPSTER and GNUTELLA have demonstrated the utility of P2P file sharing systems using various architectures: NAPSTER using a hybrid approach with a central server for maintaining the network structure while GNUTELLA implements a pure P2P network.

P2P networks not only allow the exchange of data but also the shared use of computational resources. So called Virtual Supercomputers try to use the latent processing capacity of idle machines to run computational tasks with low priority on a large number of P2P nodes. The most famous of these is the SETI@HOME program. Using P2P systems for collaboration purposes (other than game playing) is a rather new approach, one example is GROOVE[8]. This approach allows users to start discussions or new projects with ease without having to undertake administrative tasks of establishing user groups or other similar infrastructure.

4. P2P Editing of Ontologies

As discussed earlier, centralized ontology servers have a number of advantages, but they might not be suited to all environments. Using a collaborative P2P network instead of a client/server system gives different features and can therefore be considered an interesting alternative to the classic ontology editing approach. Advantages accumulate from simple administrative issues that can be avoided: to run a P2P network only client software is required. Assuming a suitable deployment mechanism, the network itself will be established by clients via scanning their network neighborhood. With the client/server approach a server has to be established and usually clients need to be configured to use the server, this means more control but also more effort.

P2P networks are by their nature de-centralized and cooperative. For ontology editing this means it is easier for people to submit changes into the ontology editing environment where they can be judged by others. It is also possible for any client to pass judgment about others' assertions into the system. In a client/server model this requires authentication and additional notification mechanisms which increase the initial editing effort. This is one of the core differences, an advantage or disadvantage, depending on the organiza-

8 *http://www.groove.net/*

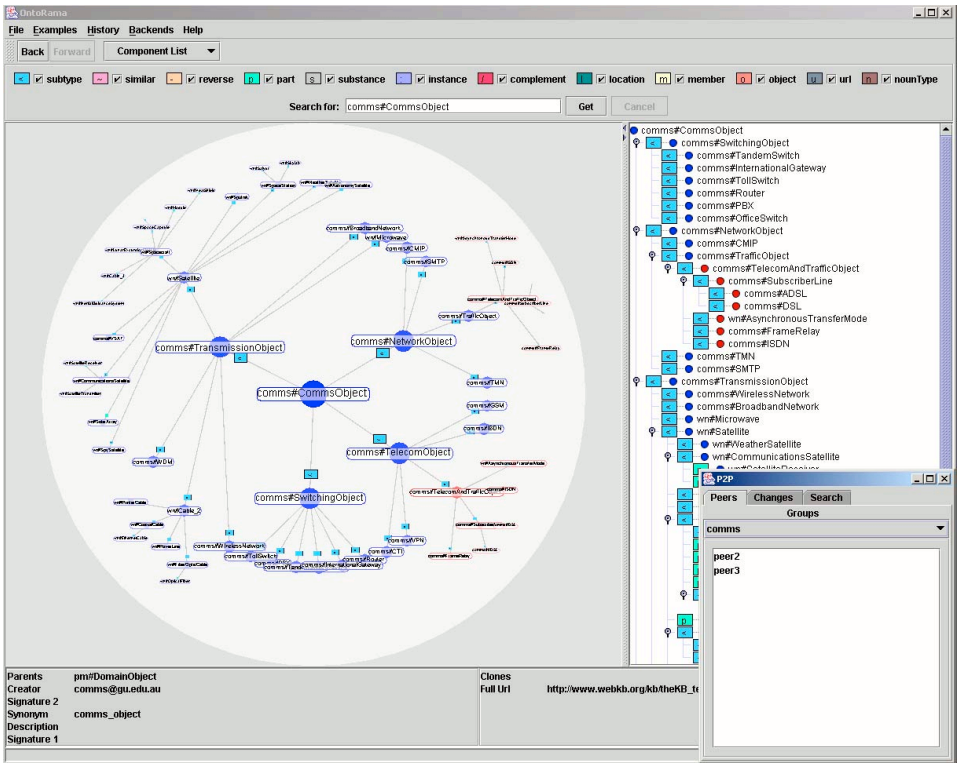

Figure 1: ONTORAMA screenshot showing main application window and *P2P Details Window*

tional context. In general, editing ontologies in a P2P network means more freedom and less control in comparison to the classical client/server approach. Changes in the P2P system can be ad-hoc – a positive trait within a organizational context of a highly collaborative environment where new ontologies are developed. However, P2P may not be a good choice for maintaining an ontology which serves other systems relying on its stability – ad-hoc changes will cause inconsistencies within the ontology. P2P ontology editing is therefore aimed towards ease of change not stability. We base our approach on RDF so an ontology can be moved to a server once it has reached a stable state for deployment.

One of the issues arising in the context of P2P collaboration is the question of how changes propagate. Directly changing the local user ontology has a number of drawbacks. Changes to non-visible parts of the ontology may be missed and changes to the visible part of the ontology might be irritating, esp. if they happen in the middle of a user operation with unexpected results. Therefore a mixed-initiative approach is more suited. The network can send notification of changes – displayed to the user – who can then accept or deny the changes and their application to his local ontology. This keeps the client up to date, but also in control of changes and the effects of their propagation.

5. Assertions and Rejections

In collaborative work users are interested in each other's opinions. Normally different opinions emerge leading to design refinement. The idea of multiple opinions or views about the way a design should proceed motivates a feature of the P2P ontology editing application that provides flexibility when adding or removing information. Rather than adding or removing statements we introduced a notion of *assertion* and *rejection*. If a client adds a node/edge to the ontology, it will be marked with an assertion statement from the client. The same approach works when removing nodes/edges – the P2P model tags an edge with a rejection statement. An example of this is shown in Figure 2 as RDF in XML notation, while Figure 3 shows the same content in graphical notation with the namespaces abbreviated for readability. Note that the P2P information is an extension of the original RDF (in RDF itself) and that the `subClassOf` relationship has been reified to allow adding the assertion and rejection statements to it.

This approach has a few advantages as opposed to simply adding & removing information. First, as mentioned above, facilitating the collaboration process. Second, a clients' nodes and edges do not simply disappear when collabora-

```
<rdfs:Class
      rdf:about= "http://www.webkb.org/kb/theKB_terms.rdf/comms#TrafficObject">
  <ontoP2P:assertedBy resource="mailto:comms@ontorama.com"/>
  <ontoP2P:assertedBy resource="mailto:john@ontorama.com"/>
  <ontoP2P:rejectedBy resource="mailto:steve@ontorama.com"/>
  <rdfs:label xml:lang="en">traffic_object</rdfs:label>
  <dc:Creator resource="mailto:comms@gu.edu.au"/>
  <rdfs:subClassOf rdf:parseType="Resource">
    <rdf:value
       resource="http://www.webkb.org/kb/theKB_terms.rdf/comms#NetworkObject"/>
    <ontoP2P:assertedBy resource="mailto:comms@ontorama.com"/>
    <ontoP2P:rejectedBy resource="mailto:marie@ontorama.com"/>
  </rdfs:subClassOf>
</rdfs:Class>
```

Figure 2: RDF fragment describing statements and their assertions and rejections.

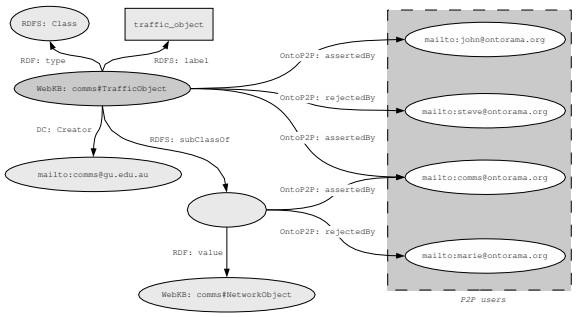

Figure 3: The RDF example shown as graph, with the additional P2P information to the right.

tors decide they don't want them. Third, the information about rejections & assertions on parts of ontology provide heuristics to ontology merging algorithms.

Therefore, within the P2P editing environment, a simple deletion of a statement is insufficient to remove it for various reasons. First, a given statement will exist on different clients, which will cause any deletion to eventually propagate back onto the client where it was initiated. To avoid this, the deletion has to be stored in some way. Further, the information – that the local user has deleted this statement – is a statement in itself and particularly relevant in the P2P context. Other users may want know about this event, and may be required to know it, so this information also has to be propagated throughout the network.

Similarly, a single user deleting a statement may not be sufficient to remove the statement from the network. Other users might not share the opinion to delete. Therefore, we implement a voting based system: every client can assert & reject statements, thereby expressing their view. This approach underlines the collaboration by adding a discursive element. Clients are identified in the system by their own personal email or URL, which can later be found in the RDF describing assertions & rejections. Assertions & rejections are then handled by two different rule sets. The first is used to decide if a statement is displayed within the local client. For example, there will often be a rule that says everything the local client rejected should not be displayed locally. Maybe the local client will want to weaken this rule to view statements he rejected, but that are overwhelmingly asserted by others. In another organizational model, an authority client might decide that this statement has to be in the ontology and therefore every client needs to see it. In this way the P2P network can be programmed to mimic the behavior of a client/server based management system.

As mentioned, if a client removes statements from his local view, this information must also to be propagated. Therefore we need to keep statements that will be not be displayed in the local client, but remain to be propagated to other clients. On the other hand, a need to remove statements across the P2P network arises, otherwise the set of statements will increase to consume network resources over time. A second rule set is therefore used to decide which statements have to be permanently deleted from the P2P network. A simple system counts assertions and rejections and removes statements rejected by the majority of clients. A more sophisticated approach adds weights to the clients, e.g. weighting assertions and rejections of the local client more strongly. Another approach is based on the current rejection status of the statement in the local client and the amount of time that has passed since any information was passed about the statement on the P2P network.

The same issues that confront automated agents on the semantic Web in the maintenance of a cache of belief statements related to RDF apply in this case. This issue, as well as in the management of central and local RDF stores, has been discussed in [13]. There is also a substantial literature on belief revision systems that can be deployed using our software framework. Obviously rules need to be adjusted for specific workgroups, e.g. some user communities prefer equal weighting of all opinions, other environments might need specific client roles to over rule others. These differences may be addressed with different sets of rules and possibly with different rule engines for client communities.

6. OntoRama

Recent developments in the Semantic Web have invigorated interest in programs that render ontological data sources (structured metadata) in the style of semantic networks used in artificial intelligence. The ONTOBROKER [7] project used a hyperbolic browser to view semantic relationships between frames in F-Logic. ONTORAMA owes its origin to the ONTOBROKER[9] browser and was adapted and re-written for use in the WEBKB-2 project [8] which uses its own internal knowledge formats (conceptual graphs) in a similar way to how ONTOBROKER uses F-Logic. A natural exten-

9 *http://ontobroker.semanticweb.org*

sion of the WEBKB-based work, with broader community interest than experimenting with knowledge annotation using conceptual graphs [8], is browsing or viewing RDF document sources. ONTORAMA is intended as a general RDF document browser but particularly focused on visualizing large structural descriptions to navigate the semantic relationships between resources. The ONTORAMA interface can be seen in Figure 1.

Rendering an ontology in the coordinate plane means that a graph should be drawn as a planar graph. Most of ontologies however are not trees, some nodes have multiple inheritance. The data structure must therefore conform to a tree and non-planar graphs transformed to this form. The solution is to copy (clone) sub-branches in the graph for vertices with multiple parents: producing a tree structure where vertices have only one incoming edge. All nodes with substructure copies are rendered red in the interface. Once the user has focused on a cloned node, all copies are also connected by an edge to enable the user to identify and unify the copies, locating them in the view. Another trait of RDF ontologies that they often contain multiple components, thereby ONTORAMA is able to browse any graph, with the various components of the graph accessible by a pull-down menu called Component list, each of nodes in this list corresponding to a root node of a tree not connected to the current view.

Ontologies may have different type relationships between the nodes, in order to visually distinguish these ONTORAMA renders a different icon for each relationship (edge). These icons have different symbols and vary in color, this makes different relations more obvious at first glance. The set of relationships is fully configurable via ONTORAMA's configuration file in order to handle ontologies coming from different sources and therefore potentially having varied sets of relationship links.

One of disadvantages of hyperbolic style view used by ONTORAMA is that if a node has many children, they become crowded and not easily read. We introduced a tree view to overcome this problem and to complement the diagrammatic view. Each view allows the user to execute the same set of commands, and if the user is focused on a substructure in one view - corresponding substructure will be focused in another view. This allows users to zoom into a node neighborhood in both views. In summary, ONTORAMA is a Java-based hyperbolic-style browser able to render ontological data as a hierarchical display. It accepts limited forms of RDF syntax.

7. Implementation

The underlying P2P system used within the modified ONTORAMA is a Java library called JXTA[10] which is used to create a pure P2P network, i.e. there are no central servers at all. In later versions an option to set up hubs for specific workgroups might be introduced, but the current version needs no specific configuration at all.

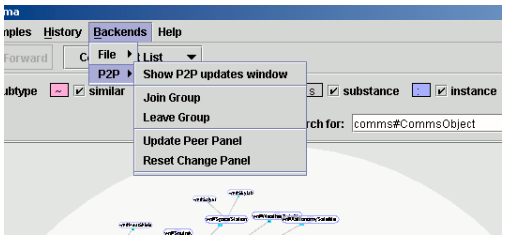

Figure 4: Backend menu in ONTORAMA

The P2P extension is integrated into ONTORAMA and we illustrate with an example of a workflow between three peers, each running a session within a P2P enabled version of ONTORAMA. After startup and loading the ontology, every user will see the ontology displayed in the spherical projection used by ONTORAMA and the other peers in his neighborhood will be shown in the *Peer Panel*, which is part of a separate window for the P2P back-end. The full application is shown in Figure 1. The ontology used in our example describes a communications terminology domain. The user has already joined a P2P group called comms with two other peers. The collaboration is initiated using a command from the P2P specific menu shown in Figure 4. New groups can be added within the dialog for joining groups.

Each user can then edit the ontology using the different views. Figure 5 shows the context menu for a node. The sub-menu opened in the figure shows a list of relations predefined for the specific ontology in an XML-based configuration file. Once a client has made a change to the ontology this information will be propagated as RDF through the P2P network and other clients receive notification of the change seen in the *Change Panel* shown in Figure 6.

The client receiving the notification can decide what to do with the changes: to accept them as they are or to reject them. In this way, a model of mixed initiative is implemented, the user herself is responsible in applying changes. The information on assertions & rejections is added to the RDF as additional statements in an ONTORAMA specific namespace. An example of such an RDF fragment was shown in Figure 2. It shows a node called TrafficObject defined within the "comms" ontology part of WEBKB-2. This node has been asserted by users identified via the email-ids comms@ontorama.org and john@ontorama.org and rejected by the user steve@ontorama.org. Additionally, the subClassOf edge connecting comms#TrafficObject and comms#NetworkObject has been asserted by the user comms@ontorama.org and

10 *http://www.jxta.org*

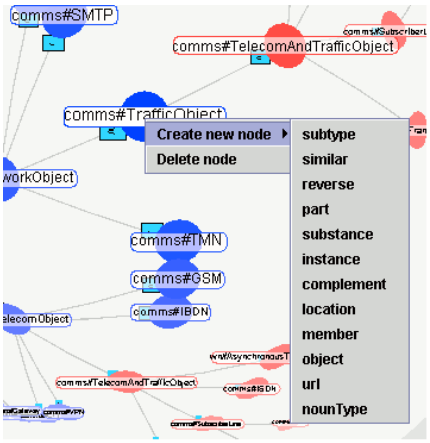

Figure 5: Popup menu allowing a user to edit ontology.

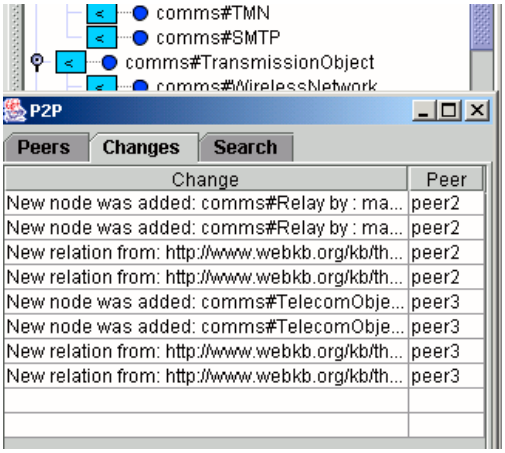

Figure 6: The *Change Panel* shows notifications about changes in the ontology coming from other peers.

rejected by user marie@ontorama.org. Whenever the information about a node changes, the complete description is sent through the P2P network. This is the same mechanism used to query the P2P network. Each peer gives a consistent set of information and the redundancies occurring when multiple changes are made to the same node.

8. Conclusion

The paper reports on a P2P solution, implemented as an extension to ONTORAMA, which allows communications with other P2P instances of ONTORAMA and potentially other ontology editing tools. The protocol is implemented using RDF and all extensions are RDF statements. This means the extended data required for the P2P solution is itself expressed as RDF. The combination of instant notification with ease of access within a P2P system provides a novel editing environment compared to the classic client/server ontology management approaches. This offers not only the implementation of new workflows in ontology editing, but also opportunities for research in the area of ontology editing, belief revision and trust systems. The open architecture of the protocol (and implementation) ease such extensions and freely available for research purposes.

References

[1] M. Arumugam, A. Sheth, and B. Arpinar. Peer-to-peer semantic web: A distributed environment for sharing semantic knowledge on the web. `http://webster.cs.uga.edu/~budak/papers/workshop02.pdf`, 2002.

[2] S. Bechhofer, I. Horrocks, C. Goble, and R. Stevens. OilEd: A reason-able ontology editor for the semantic Web. *Lecture Notes in Computer Science*, 2174:396–406, 2001.

[3] Y. Ding. A review of ontologies with the semantic web in view. *Information Science*, 27(6):377–384, 2000.

[4] J. Domingue, E. Motta, and Corcho Garcia. Knowledge modelling in webonto and ocml: A user guide. `http://kmi.open.ac.uk/projects/webonto/`, 1999.

[5] P. Eklund, R. Cole, and N. Roberts. Visualizing and retrieving ontologies. In S Staab and R. Studer, editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 405–414. Springer-Verlag, 2004.

[6] A. Farquhar, R. Fikes, and J. Rice. The ontolingua server: A tool for collaborative ontology construction. Technical report, Stanford KSL 96-26, 1996.

[7] D. Fensel, S. Decker, M. Erdmann, and R. Studer. Ontobroker: Or how to enable intelligent access to the www. In *Proc. 11th Knowledge Acquisition Workshop (KAW98)*, pages 8–23. Banff, Canada, 1998.

[8] P. Martin and P. Eklund. Embedding knowledge in web documents. In *The Eighth International World Wide Web Conference, (WWW8)*, pages 324–341. Elsevier, 1999.

[9] P. Mitra, G. G. Wiederhold, and M. Kersten. A graph-oriented model for articulation of ontology interdependencies. `http://www-db.stanford.edu/pub/gio/2000/ONION.pdf`.

[10] 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*, 2(16):60–71, 2001.

[11] Y. Sure, M. Erdmann, J. Angele, S. Staab, R. Studer, and D. Wenke. OntoEdit: Collaborative ontology development for the semantic web. In *Proceedings of the first International Semantic Web Conference 2002 (ISWC 2002), June 9-12 2002, Sardinia, Italia*. Springer, LNCS 2342, 2002.

[12] Peer to Peer Working Group. What is peer-to-peer? `http://www.p2pwg.org/whatis/index.html`, 2002.

[13] M. Zhurakhinskaya. *Belief Layer for Haystack*. PhD thesis, MIT, 2002.