Easing Participation in the Semantic Web

Stefan Haustein, Jörg Pleumann
Computer Science VIII, X
University of Dortmund,
Baroper Str. 301
D-44221 Dortmund, Germany
{stefan.haustein, joerg.pleumann}@udo.edu

ABSTRACT

Although a promising idea, the Semantic Web currently seems to have a problem duplicating the success story of its predecessor, the World Wide Web. The number of people actively participating in the Semantic Web has been very limited until now, because people can't see the benefits originating from the extra effort they invest into semantically rich web pages. Unfortunately, this advantage is barely visible at all until a critical mass of RDF-annotated pages is available on the net, thus making is difficult to recruit new participants for the Semantic Web. The article tries to break this vicious circle by showing that the use of appropriate tools may both ease participation in the semantic web and provide a number of additional advantages not directly related to the Semantic Web. The latter, in particular, may convince a larger number of people to participate, and thus bring the Semantic Web nearer its critical mass.

1. INTRODUCTION

The Semantic Web is a great idea. Yet, it did not quite take off until now. Why is this the case? Some argue that RDF [19], the language for adding the semantic information to existing web pages is the problem. These critics see RDF as being too complicated or under-specified [11, 6]. While RDF truly has its problems in some areas, we don't think that the language itself is the main obstacle that hinders people from participating in the Semantic Web. But to find out where the problem actually lies, we first need to take a step back and look at what made the original web such a tremendous success.

In our opinion, there were four important reasons for the success of the World Wide Web:

- Simplicity. HTML was easily understood and quickly written down. Even novices could design a few basic web pages with little effort, put them in a matching directory structure and start an HTTP daemon to deliver the content to clients.

Semantic Web Workshop 2002 Hawaii, USA

- Immediate feedback. After an HTML page had been designed in a text-editor, the result could be displayed in any HTML client to get an impression of the results. Thus, the user had an immediate feedback on his or her work.

- Additional benefits. Even though their original purpose was to present information to other people, HTML pages could be used as a means of discussion or documentation for people participating in a project or even for personal use. Thus, there was an additional gain users got from participating in the world wide web, which made the system even more attractive to them.

- Low critical mass. As a networked effort, the World Wide Web required a minimum (but large enough) number of participants to raise the interest of outside people, convincing them to become involved. Yet, since the World Wide Web was the first system of its kind, and there was no similar system to compete with, this critical mass was relatively low.

When we compare these points to the Semantic Web in its current form, we notice that most of them are not fulfilled:

- Simplicity is only partially given. The mixture of RDF and DAML+OIL is understood in all its details only by people that have a background in AI or related fields. Novices will only be able to use basic concepts of RDF and might thus have problems to see the real advantages of the Semantic Web.

- Immediate feedback is not given. Unfortunately, there is no specific client software for the Semantic Web that gives users an impression of their RDF fact base. One could argue that it doesn't even make sense to ask for such a software, because the clients of the Semantic Web are programs rather than human beings.

- There are no additional benefits, at least none that are ovious to "'ordinary end-users"'. While human-readable HTML pages primarily designed for other people can also be used for personal purposes, this is not true for RDF facts, which are meant to be read by programs.

- The critical mass is considerably higher. Why is this the case? This time, there already is an existing system — the original World Wide Web — , and

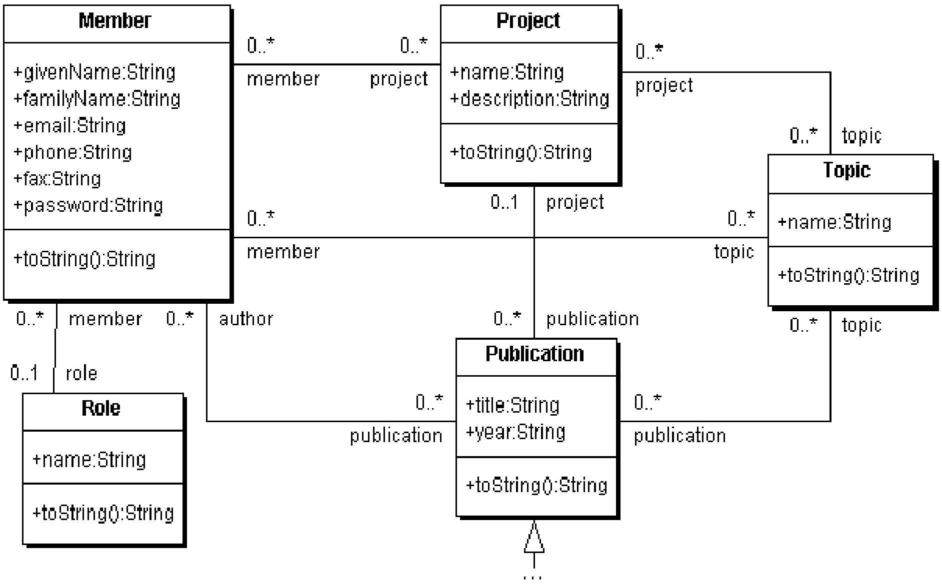

Figure 1: Simple UML Diagram for university department's web site

most people nowadays tend to use the ”‘brute force”’ method to find a specific piece of information in it, namely Google or some other search engine. Thus, it is more difficult to convince people to take part in another system, even if it is an extension to the existing one.

As long as the first three points are true, the critical mass of users needed to make the Semantic Web ”‘take off”’ will be hard to reach. Unfortunately, seen the other way round, the Semantic Web hardly has some kind of real benefit unless there is a large-enough number of participants that makes available RDF-specified information to others, that is, until the critical mass is reached. The current situation could be seen as some kind of vicious circle that has to be broken before the Semantic Web has a chance to succeed.

2. TOOLS TO BREAK THE CIRCLE

To break the circle, we have to get rid of as many as possible of the four problems shown in the previous section. Since we cannot lower the critical mass for mainstream acceptance of the Semantic Web (possibly by forcing people into it), we have to focus on the other three: Simplicity, immediate feedback, and additional benefits. A very promising way to achieve this seems to be the use of appropriate tools. These tools would have to ease participation in the Semantic Web, but would also have to provide some ”‘added value”’ that makes them attractive to end-users. Obviously, when using the tools, people will also likely participate in the semantic web, even if that is not their original motivation. The following sections try to show what features these tools might offer.

2.1 Generative approach

Looking at existing tools developed for or related to the Semantic Web, for example Protégé-2000 [23] or Ontobroker [15], one notices that these are primarily designed to support ontology and fact management. Information is stored in a knowledge base providing fine grained access. The ontology is utilized to make sure that the content corresponds to the desired structure. The two systems mentioned above are able to export their fact base to an RDF representation.

While these tools aim into the right direction, they still have a problem: As long as one wants a machine-readable RDF-version of the facts as well as a human-readable HTML-version, duplicate effort is required to maintain both. Take, for example, a typical web site for a university department containing information about the department's staff, research topics, projects, and publications. A highly structured site like this is suitable for participating in the semantic web, and it can easily be modelled using a corresponding domain ontology. Yet, a change as simple as a telephone number has to be propagated to the RDF version as well as the HTML version.

Given a Semantic Web tool followed a generative approach, the situation would be easier: Assume this tool were able to incorporate regular HTML for the unstructured part of the web site, and these pages could contain placeholders for insertion of information contained in the fact base. The tool would then be able to generate the actual HTML pages automatically from the existing RDF information – or even both from a common fact base – , thus requiring the user to maintain this fact base only, at least as far as structured information is concerned. If the generation of pages takes place at run-time, we arrive at a tool that could be seen as a ”‘Semantic Web-enabled HTTP server”’

While the avoidance of redundancy already is a big advantage addressing simplicity, the generative approach provides other advantages that fall into the area of ”‘added value”’:

- In contrast to editing HTML directly, a unique look and feel can easily be established for the whole site, given an appropriate template mechanism.
- In addition to HTML and RDF, other target formats

like WML and cHTML can be generated from the same fact base, lowering redundancy even further.

- In contrast to plain HTML files, ontology-based consistency checks can be performed automatically while entering data, e.g. avoiding dangling links inside the system.

2.2 Incorporation of database features

To broaden the possible target audience of our Semantic Web server, we might try to incorporate database-like features and thus position it as an alternative to a "'heavy-weight"' database solution.

While relational databases with HTML-generating front-end are quite common these days (e.g. Cold Fusion [8], PHP [2], Enhydra [1] etc.), these solutions are mainly used for sites with a simple, low-dimensional structure, such as guest books or news pages (e.g. Slashdot.org). More complex domains such as university departments often still use plain HTML files for their web presentation, or make only limited use of database tables.

Here, the reason may be that a high number of tables would be required for modeling even simple ontologies, mainly because associations are not first class members of relational database systems. Revisiting the university department scenario, we need at least tables for persons, research topics, projects, and publications. Figure 1 shows a possible UML class diagram of the database's conceptual model. Since all n:n associations require separate association tables, this results in quite a lot of normalised tables (more than 10), each of which potentially contains only a very small subset of all the possible instances.

In this case, the benefit for the creator, that is, the dynamic generation of HTML or – in our case – RDF from a single set of data, does not outweigh the extra effort inherent in maintaining the tables.

Using Semantic Web tools, the picture may change significantly. For a low number of instances, the internal knowledge base provided by a Semantic Web tool may be sufficient. Associations are direcly supported, and the ontology language also allows to specify integrity constraints for them at an appropriate level. Since Semantic Web tools usually come with a generic user interface, the need to create HTML forms for editing the tables is avoided.

2.3 Incorporation of Content Management Features

Another area that a Semantic Web tool might address is content management. Content management systems, such as Hyperwave [3], Zope [5] or OpenCMS [4] provide user, version and metadata management for a set of HTML pages or binary documents in other formats such as PDF or Word. Their set of meta data, hoewever, is usually fixed and tailored to the most common needs. Here, ontology-based Semantic Web tools provide much more flexibility, and may be superior to general content management systems in domains where the meta data requirements significantly differ from the standard set provided by content management systems.

2.4 Openess to Alternative Schema Languages

In the introduction, we claimed that beneath providing no gain that becomes immediately obvious, RDF annotation is complex.

In its current form, the Semantic Web requires users to learn yet another formal description language. Users having an background in AI may be expected to be familiar with description logics and corresponding ontology modelling tools. For mainstream acceptance, though, integration of recognised standards like UML [20] may help to improve acceptance of Semantic Web tools and thus lower the entrance barrier [13]. Most students of computer science or related engineering disciplines can be assumed to be familiar with UML and modelling tools like Together or Rational Rose. These students could easily apply their modelling knowledge to the Semantic Web and thus contribute to its group of early adopters.

3. THE INFORMATION LAYER

In order to demonstrate that participation in the Semantic web actually can be simple, and that using a server based on a fine grained fact base instead of HTML- or RDF files can provide immediate gains, we have started to model our own unit's web pages accordingly. For this purpose, we used our Information Layer system, which stores data in a simple XML format that is determined by a given ontology. The information layer uses an object-oriented model for data representation. Objects consist of atomic attributes and relations to other objects. The consistency of relations in both directions is ensured automatically, avoiding inconsistencies inside the system. The concepts and relations are defined application-dependent in an external ontology definition file. All files used by the information layer are stored as XML documents.

The InfoLayer system was originally designed as an integrated information platform for software agents and human users in a conference scenario. The system was used in the COMRIS project [21] in order to make conference information available in appropriate formats to human users as well as software agents, utilizing the same underlying knowledge base. Access to the content is possible via a generic HTML interface as well as a FIPA [16] based XML interface [18]. Obviously, when information is machine readable for software agents, it is not a big leap to make this information available for the Semantic Web as well.

In the process of modelling our unit web pages, we made several improvements to our system, simplifying the use as a replacement for a "regular" web server. While there may be alternative paths appropriate for other systems, our main purpose was to show that using semantic web systems may provide direct advantages over regular web servers, even without relying on advanced features such as knowledge integration from different sources (e.g. KAON-REVERSE [17]).

3.1 XMI Import

The original version of the Information Layer system used its own proprietary XML-based ontology description language. In order to simplify the initial step of generating the application ontology, we have replaced the internal format by XMI [20], the XML based exchange format for UML diagrams. Figure 1 shows a simplified version of the UML model currently used as a basis for our unit web pages.

We have chosen UML as ontology modelling language [13] instead of RDFS [7] because it is difficult to avoid contact with UML when working in computer science or in the IT industry in general. For most computer scientists, a UML editor like Rational Rose or Together is part of their standard

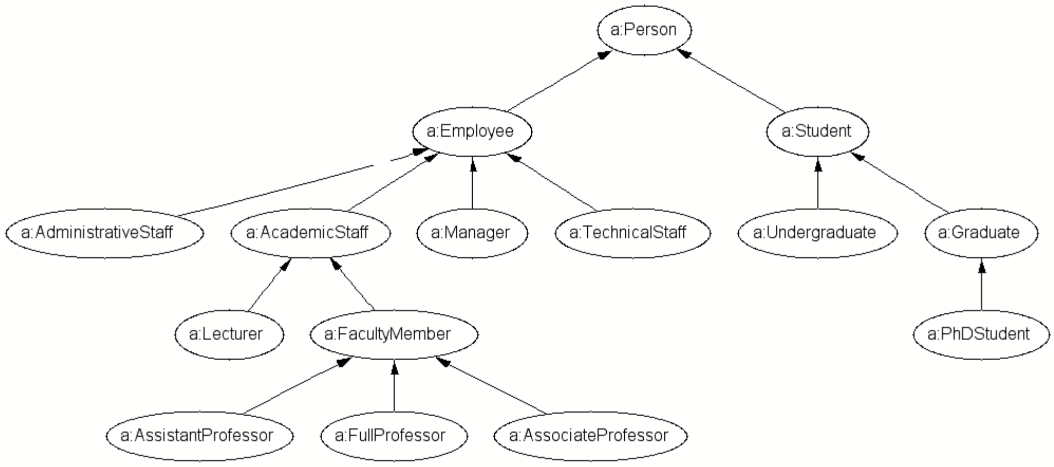

Figure 2: A Subset of the Semantic Web Research Community ontology concept Hierarchy

tool box. Thus, the extra effort of installing and getting familiar with an RDFS editor, possibly preventing people from getting in touch with the Semantic Web, is avoided.

Compared to other languages suitable for ontology modelling, UML currently still lacks clearly defined semantics. However, there are significant efforts to solve this problems [22, 10].

This aspect may be less important for systems providing their own comfortable Ontology editor.

3.2 HTML Generation

The most important capability required for being able to replace existing web servers is – of course – the generation of HTML pages.

The information layer contains a module that provides built-in web-server functionality. The server is able to generate HTML dynamically: For any object, the attributes are simply displayed, and the associations to other objects are converted to sets of hyperlinks to the related objects. Concepts are displayed as a clickable list of instances corresponding to the concept. The HTML interface can also be used to edit the content of the system using forms generated dynamically based on the ontology. In the COMRIS project, the HTML interface was used for interaction with the end user as well for as debugging and inspection purposes.

In addition to generic HTML generation, templates can be used in order to generate HTML pages conforming to a given look and feel. In the COMRIS project, we have also used the template mechanism to generate the input structure required by the text generation system TG/2 ([9]) which was used to generate natural language output for a wearable device. The template mechanism is described in some more detail in the next section.

3.3 SWRC and RDF Integration

The Semantic Web Research Community (SWRC) Ontology [24] is an ontology designed in order to describe the structure of the Semantic Web Research Community, namely the members, events, topics and projects, in a machine-readable manner. It is available in DAML+OIL and FLogic formats. Figure 2 shows a subset of the inheritance hierarchy of the SWRC ontology.

Since our "local" research unit ontology was primarily designed to fit the needs of our "regular" web presentation, it does not match the "shared" SWRC ontology exactly. However, using the template mechanism of our system, we are able to generate RDF pages corresponding to the SWRC ontology on the fly. Figure 3 shows a simplified example template that is used to generate SWRC-compliant RDF content for instances of the class "Member". In the templates, elements in a special namespace, denoted by the t prefix in the example, are replaced by content queried from the Information Layer with respect to the current instance which is determined from the page URL.

Thus, it is possible to participate in the Semantic Web without needing to extend a predefined shared ontology, which may be bloated and still not fullfill all local requirements. Instead, the domain of interest can be modelled using a lean domain specific local ontology. The SWRC person name slot illustrates the advantage of this approach: SWRC contains only one person name slot that is not split into first and last name. If the local application requires having both parts available separately, it would be necessary to duplicate the corresponding information, when building the local ontology on top of the SWRC ontology. Also, SWRC concepts like "Organization" may not be required in a local ontology covering a single organization. Information about the local organization can be stored in a single static RDF file, not bloating the local ontology.

In addition to template based RDF generation, it would be possible to generate RDF directly corresponding to the local ontology automatically [12]. However, this feature is not implemented yet.

3.4 Infrastructure Integration

For simpler integration with the existing Web server infrastructure, we changed the Information Layer implementation to become a Java Servlet instead of a stand alone program. Running the Information Layer as a Java Servlet allows smooth integration with existing Web presentations, without any hard switch. The service can simply be added where it makes most sense, and then later be extended to

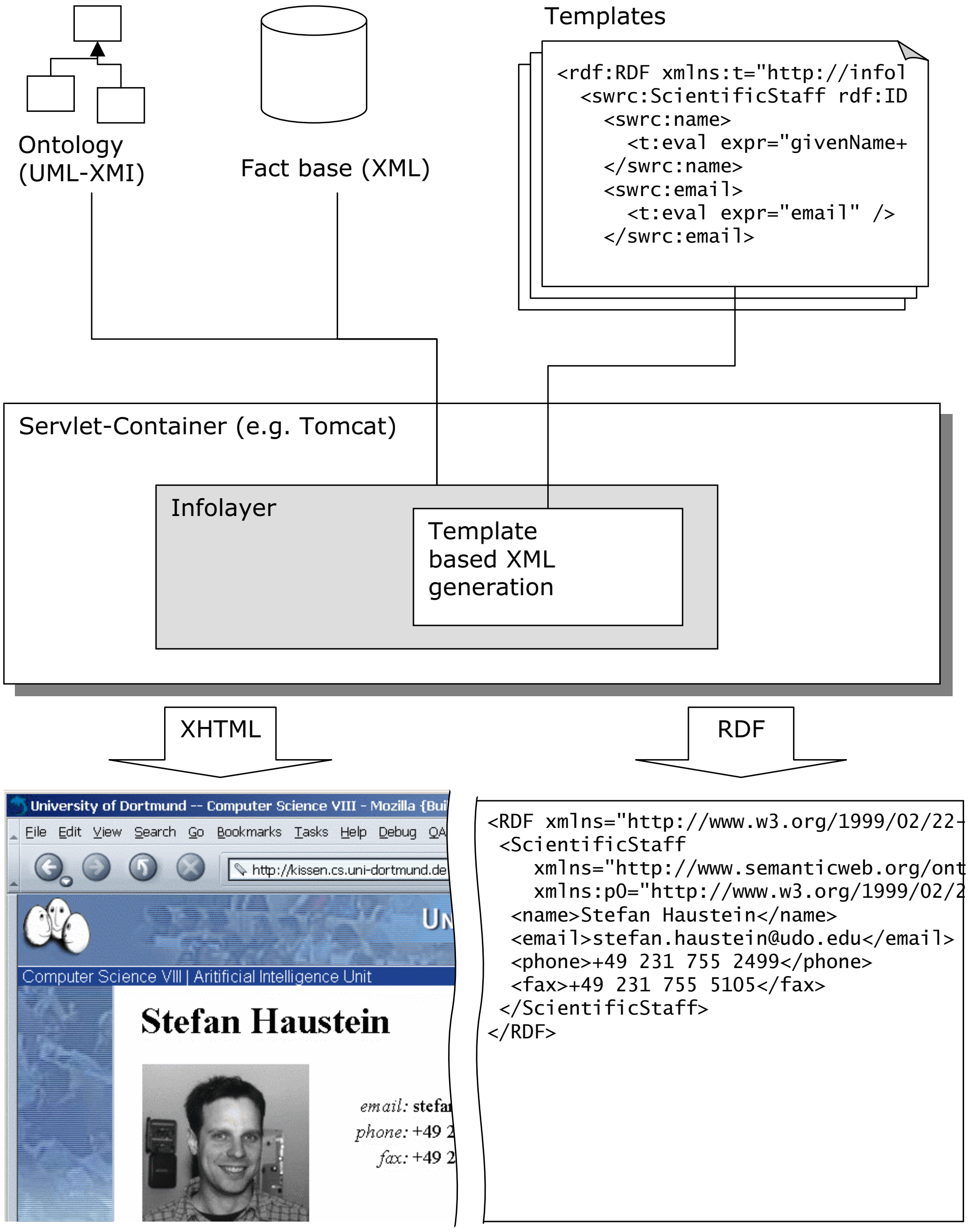

Figure 3: Dynamic HTML and RDF generation using the Information Layer template mechanism. Elements prefixed with a t: are evaluated with respect to the current instance as given in the page URL.

other areas.

3.5 File Upload

Last but not least, we have the option to upload arbitrary Files (PDF, MPG, ...) into the system. We have added this feature in order to improve suitability for general useage. While it may look a bit odd here on the first sight, it is a typical feature of content management systems. Of course, the content of the files is opaque to the system, which is controversial to the idea of providing fine grained information in RDF-format. However, the system supports the addition of relevant meta-information.

3.6 Installation

A complicated installation procedure may prevent potential users from actually using a system, even if there are obvious time or cost savings in the long run. Building on the system improvements described above, the installation of an Information Layer based system was reduced to the following steps:

1. Build a simple base ontology with the UML tool of your choice, or just use the sample ontology available from the infolayer web page as a starting point.

2. Install Apache-Tomcat or any other Web server that is capable of handling Java servlets, if not already available.

3. Install the Information Layer Servlet files in the Web services directory of the server and adopt the configuration in the `web.xml` file to your local environment.

Following these steps, a user is already able to add content using the generic Web interface and to view the content using that interface. Now the system can be further enhanced, by extending the ontology and by adding XHTML and RDF templates, customising the look and feel and the RDF generation properties of the system.

Please note that the temporal frame of the latter two steps is not fixed. For example, one could start with managing publications using the system, and then later add other concepts like projects, topics, persons or courses.

3.7 Related Systems

Obviously, other Semantic Web tools may be extended similarily. Protégé-2000 is a Knowledge Base supporting RDF format. It provides a nice Java user interface including an ontology editor, but currently lacks a plain HTML interface. Other semantic web tools such as Ontology Builder [14] or the KAON framework [17] seem to focus more on enterprise-level ontology management and information integration.

4. CONCLUSION AND OUTLOOK

The Semantic Web is a great vision. However, for a broad adoption, simple tools that allow participation without a background in AI are still rare. Protégé and similar tools seem to aim in this direction. We would like to contribute our own tool, the Information Layer. While other tools focus on easing the ontology building process, we mainly tried to address simplicity in the overall system as well as providing additional benefits that might persuade users to participate in the Semantic Web. One of these benefits is the generation of HTML as well as RDF from a common fact base to avoid redundancy, others are the incorporation of database and content management features to broaden the target audience. This way, we hope to improve availability of structured information suitable for the Semantic Web. We did not put a focus on advanced features like full DAML+OIL support, nor do not have a priority here in the future.

A web site that utilizes the Information Layer in its current form is a database for Java-enabled small devices like cell phones and personal digital assistants[1]. Here, the ontology descibes the devices, their capabilities, vendors, available protocols and known bugs. Changes to the fact base are quite frequent, but do not require the duplicated effort of updating a human and a machine-readable version, which makes the site very easy to maintain.

The Information Layer is also being used as a prototypical web presence for MuSofT, a Germany-wide project that develops multimedia teaching material for software engineering education. The site's goal is to manage and distribute the learning objects contributed by the various project partners. This installation makes use of the content management features the system provides: Learning object can be uploaded into the system from a Web browser. To allow efficient retrieval of material, LOM[2]-conforming meta-data is provided using the system's ontology capabilities.

Previous versions of the Information Layer system have been and still are are used as a basis for the MLnet teaching information server[3] and in various internal projects.

For more details about the Information Layer and its applications, please refer to `http://infolayer.org`.

5. REFERENCES

[1] The home of enhydra.org, 2001. http://www.enhydra.org.

[2] Php: Hypertext preprocessor - homepage, 2001. http://www.php.net.

[3] Hyperwave homepage, 2002. http://www.hyperwave.com.

[4] Opencms homepage, 2002. http://www.opencms.org.

[5] Zope homepage, 2002. http://www.zope.org.

[6] Dan Brickley. RDF interest group - issue tracking. Technical report, World Wide Web Consortium, 2000. http://www.w3.org/2000/03/rdf-tracking/.

[7] Dan Brickley and R. V. Guha. Ressource Description Framework (RDF) Schema specification 1.0. Technical report, World Wide Web Consortium, 2000. http://www.w3.org/TR/CR-rdf-schema-20000327.

[8] Rob Brooks-Bilson. *Programming ColdFusion*. O'Reilly, 2001.

[9] Stephan Busemann. A shallow formalism for defining personalized text. In *Workshop Professionelle Erstellung von Papier- und Online-Dokumenten at the 22nd Annual German Conference on Artificial Intelligence (KI-98)*, Bremen, September 1998.

[10] T. Clark, A. Evans, S. Kent, S. Brodsky, and S. Cook. A feasibility study in rearchitecting UML as a family of languages using a precise OO meta-modeling

[1] http://www.kobjects.org/devicedb

[2] http://ltsc.ieee.org/wg12/index.html

[3] http://kiew.cs.uni-dortmund.de:8001/

approach. Report, Precise UML Group, September 2000. http://www.cs.york.ac.uk/puml/mml/mmf.pdf.

[11] Wolfram Conen, Reinhold Klapsing, and Eckhart Koeppen. RDF model and syntax revisited: From reification to nesting, from containers to lists, from dialect to pure xml. In *International Semantic Web Working Symposium (SWWS)*, pages 195–208, Stanford University, California, USA, 2001.

[12] S. Cranefield. Networked knowledge representation and exchange using UML and RDF. *Journal of Digital information*, 1(8), 2001.

[13] S. Cranefield and M. Purvis. Uml as an ontology modelling language. In *Proceedings of the Workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence (IJCAI-99*, 1999.

[14] Aseem Das, Wei Wu, and Deborah L. McGuinness. Industrial strength ontology management. In *International Semantic Web Working Symposium (SWWS)*, pages 17–37, Stanford University, California, USA, 2001.

[15] Stefan Decker, Michael Erdmann, Dieter Fensel, and Rudi Studer. Ontobroker: Ontology based access to distributed and semi-structured information. In R. Meersman and other, editors, *Semantic Issues in Multimedia Systems, Kluwer Academic Publisher, Boston, 1999*. Kluwer Academic Publisher, Boston, 1999.

[16] Foundation For Intelligent Physical Agents (FIPA). *FIPA Agent Management Specification*, 2000. http://www.fipa.org/specs/fipa00023/XC00023F.pdf.

[17] Siegfried Handschuh, Alexander Maedche, Ljiljana Stojanovic, and Raphael Volz. KAON - the KArlsruhe ONtology and Semantic Web infrastructure. http://kaon.semanticweb.org, 2002.

[18] Stefan Haustein. Utilising an ontology based repository to connect web miners and application agents. In *Proceedings of the ECML/PKDD Workshop on Semantic Web Mining*, September 2001.

[19] Ora Lassila and Ralph R. Swick, editors. *Resource Description Framework (RDF) Model and Syntax Specification*. World Wide Web Consortium, 1999. http://www.w3.org/TR/1999/REC-rdf-syntax-19990222.

[20] Object Management Group. OMG Unified Modeling Language Specification, version 1.3. http://www.omg.org/technology/documents/formal/unified_modeling_language.htm, 2000.

[21] Enric Plaza, Josep Llus Arcos, Pablo Noriega, and Carles Sierra. Competing agents in agent-mediated institutions. *Personal Technologies Journal*, 2(3):1–9, 1998.

[22] Precise UML Group. The Precise UML Group home page. http://www.puml.org, 2001.

[23] Stanford University. *Using Protégé-2000 to Edit RDF*, June 2000. http://protege.stanford.edu/protege-rdf/protege-rdf.html.

[24] York Sure. Swrc - semantic web research community ontology, 2001. http://ontobroker.semanticweb.org/ontos/swrc.html.