SemanticWeb.org - A Collaborative Portal for the Semantic Web Community

Benjamin Nowack[1], Andreas Harth[2], Stefan Decker[2]

[1] appmosphere web applications, Essen, Germany
bnowack@appmosphere.com

[2] DERI - Digital Enterprise Research Institute, Galway, Ireland
{andreas.harth, stefan.decker}@deri.org

Abstract

Created in the year 2000, the Semantic Web Community Portal is still one of the first Web sites visited by people looking for information about the Semantic Web. However, due to both technical restrictions and limited resources content updates became rare and were finally discontinued in 2003. This paper describes the infrastructure for a new SemanticWeb.org site which is based on the Resource Description Framework (RDF). The new system will allow the whole Semantic Web community to contribute to the portal and also to repurpose the jointly maintained data.

1 Introduction

The Semantic Web Community Portal[1] was created in early 2000 and has since then been one of the main entry points to information about Semantic Web events, projects, tools, companies, and related technologies. The site was based on manually edited static pages, quick updates to the portal became increasingly difficult with a growing amount of information. Limited resources to maintain the portal led to the discontinuation of content updates in 2003.

In the meantime, the Semantic Web and the number of interested researchers and Web developers have evolved, thus increasing the need for a community portal. This poster presents the RDF-based portal framework which was used to relaunch the SemanticWeb.org site.

2 Main Objectives

We were aiming at a Semantic Web portal both content- and technology-wise. The site should offer information *about* the Semantic Web, but it should also be *based on* Semantic Web technologies.

2.1 A "Semantic Web" Portal

From the content perspective, we decided to cover:

- Semantic Web events (conferences, workshops, ...),
- People (researchers, developers, organizations, ...),
- Projects (general projects, software projects, online services),
- Developer Tools (RDF toolkits, editors, reasoners, ...),
- End-user applications (PIMs, calendars, ...),
- Publications (articles, books, ...).

The portal is meant to address not only researchers, but also implementors, technology journalists, or people who heard about the Semantic Web but don't neccessarily have any detailed background knowledge. The goal of the portal is to help the Semantic Web spread by providing information for all audiences.

2.2 A "Semantic" Web Portal

Concerning the technical implementation, we wanted to:

- aggregate and refresh information from various sources,
- allow users to conveniently annotate resource descriptions without the need to download or install any software in order to build a large group of contributors ("on-site contribution"),
- allow users to integrate remotely maintained resource descriptions in the portal ("off-site contribution"),
- enable the repurposing of locally stored information,
- build an open system with an extensible data model.

Besides the basic requirements, we also wanted the portal to become a Semantic Web showcase.

3 Implementation Approach

The portal is based on a combination of commercial[2] and open source[3] components created by one of the project

[1] http://www.semanticweb.org/

[2] http://www.appmosphere.com/en-owlchestra
[3] http://www.appmosphere.com/en-arc

members. This allowed us to direct the development of the system's infrastructure along practical requirements without the need to depend on extensions of a third party toolkit, and without having to write middleware in order to connect different tools.

Conceptionally, SemanticWeb.org differs from most current RDF-enhanced sites as we tried to model an integrated ontology from a range of widely deployed vocabularies instead of creating a portal-specific term set. The initial ontology consists of terms from more than 25 namespaces with the ability to extend the model at run time via a Web browser. The ontology entries and axioms are stored in a relational database, so that an OWL API can be used to enable basic inferencing functionality such as class tree expansion or identity reasoning.

Basing the whole system on PHP[4] facilitated the iterative development. Although interpreted scripting languages are less performant than pre-compiled code, we managed to keep response times acceptable by using three different repositories for the ontology, user-maintained annotations, and public RDF data.

4 Architecture and Portal Components

Figure 1 shows the portal's architecture. The system can generate both human- and machine-readable results. Each application is Web-accessible and can utilize a set of APIs.

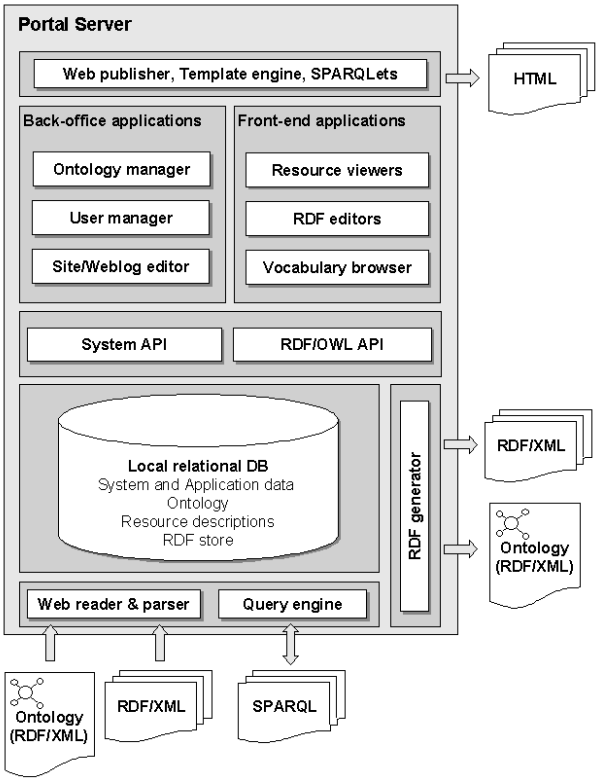

Figure 1: System architecture

[4] http://www.php.net/

Among the (human-oriented) back-office applications is a "classic" site editor to create pages and editorial content, a user manager, and an ontology editor. The front-end consists of mainly public applications such as viewers and browsers for the different types of resources. Registered users get access to annotation and RDF editing tools. HTML-serving is done via the system's Web publisher, a template mechanism to render complete Web pages, and interactive page items generated from the RDF store ("SPARQLets"). Unknown resources can be displayed by a generic RDF browser, resources supported by the local ontology are rendered in optimized, type-specific viewers.

An HTTP reader can import RDF individuals or ontologies from the Web. We also created some wrappers to be able to include non-RDF data.

Resource description export and retrieval is made possible through an RDF generator, support for URIQA requests [Stickler, 2003], and a basic data access interface for remote SPARQL (SPARQL Protocol And Query Language) [Clark, 2005; Prud'hommeaux and Seaborne, 2005] queries.

5 Lessons Learned

The already available RDF-related specifications are stable and mature enough to implement feature-rich Semantic Web applications. But we discovered that developing a system suitable for people who are not RDF experts requires means to partly restrict the Resource Description Framework's openness. There is still a trade-off between extensibility and usability. However, by utilizing the underlying OWL model, and by partly falling back to custom restrictions, it seems to be possible to combine RDF's flexibility with user-friendly browsers and editors and to return consolidated, almost exact results. The current system still needs to be improved, but it already demonstrates functionality which wouldn't be possible without the open nature of RDF.

Acknowledgements

Benjamin Nowack would like to thank DERI and especially Stefan Decker for the great opportunity to work on this project, for the guidance, and his patience. He would like to thank Andreas Harth for both technical and conceptual help, and Kerstin Zimmermann for various helpful comments.

References

[Stickler, 2003] Patrick Stickler. *URIQA: The Nokia URI Query Agent Model*.

http://sw.nokia.com/uriqa/URIQA.html

[Clark, 2005] Kendall Grant Clark. *SPARQL Protocol for RDF*.

http://www.w3.org/TR/rdf-sparql-protocol/

[Prud'hommeaux and Seaborne, 2005] Eric Prud'hommeaux, Andy Seaborne. *SPARQL Query Language for RDF*. http://www.w3.org/TR/rdf-sparql-protocol/

SemanticWeb.org - A Collaborative Portal for the Semantic Web Community

ISWC 2005 Demo Explanation

1 Introduction

The poster "SemanticWeb.org - A collaborative Portal for the Semantic Web Community" will be accompanied by a live demonstration of the system's back-office and front-end applications. Additionally, we will provide information about architecture decisions and innovative features such as SPARQL-driven portlets or the portal's interactive RDF editing forms.

2 Components to be demonstrated

Integrated Ontology Manager

Many of the portal's advanced features are based on the underlying OWL model. But instead of using an external ontology editor, SemanticWeb.org is built around an integrated vocabulary manager (Figure 1), so that the ontology can be maintained and extended at run-time by multiple authors. Synchronizing different model versions is not neccessary as the descriptions of classes, properties, datatypes etc. are kept on the server.

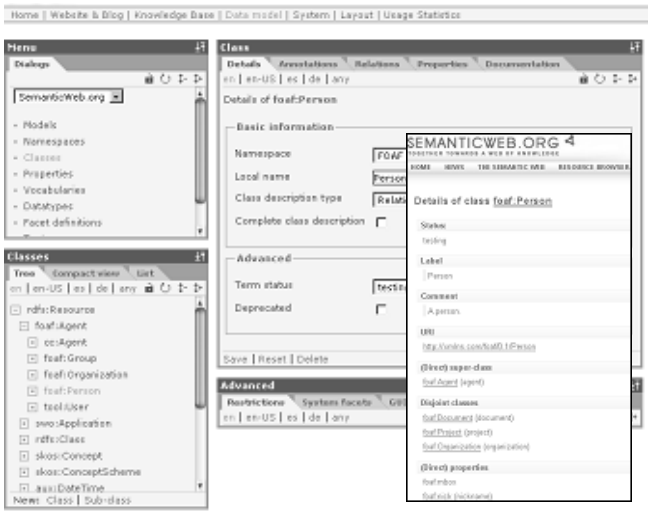

Figure 1: Ontology manager and generated class summary

The Web-based vocabulary manager is also used as a vocabulary server for both Semantic Web agents and human users. Additionally, it offers the possibility to assign validation functions to datatypes and properties and the mapping of custom datatypes to user interface widgets.

Web-based RDF editors

The portal provides an interactive, browser-based editing environment for RDF instance data. Forms can be auto-generated and/or augmented with information from the ontology (tooltips, form field labels, drop-down lists, etc.) as shown in Figure 2.

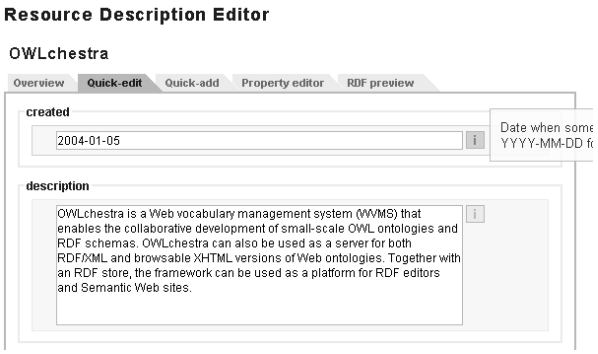

Figure 2: OWL-backed RDF editing form

The editor supports collaborative editing and can be extended with custom form templates.

Interactive "SPARQLets"

For the creation of various portlets we developed page items generated from SPARQL queries. These SPARQLets can be refreshed independently of the rest of the page. Combined with multi-tab boxes, they allow the integration of a large amount of information in a single page while keeping response times low and pages clearly arranged (Fig. 3).

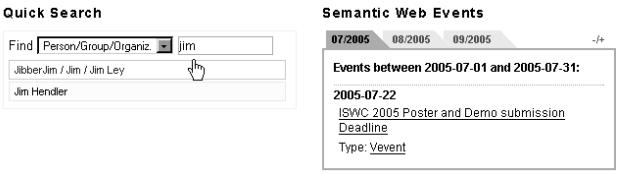

Figure 3: SPARQLets and multi-tab boxes