Publishing and Using Ontologies as Mash-Up Services

Kim Viljanen, Jouni Tuominen, and Eero Hyvönen

Semantic Computing Research Group (SeCo)
Helsinki University of Technology and University of Helsinki
P.O. Box 5500, 02015 TKK, Finland
first.last@tkk.fi, http://www.seco.tkk.fi

Abstract. The Semantic Web is based on using ontologies for enabling semantically disambiguated data exchange between distributed systems on the web. This requires efficient means for publishing ontologies on the web to ensure the availability, sharing and acceptance of the ontologies. Support services are needed for utilizing ontologies easily and cost-effectively in applications and legacy systems lacking ontology support. To address these vital needs, this paper presents the ONKI ontology service which provides ready-to-use "mash-up" functionalities, such as semantic disambiguation, concept finding and concept fetching as ready-to-use web widgets for adding ontology support to e.g. HTML forms using JavaScript. Two implementations of the ONKI Server are presented: ONKI-SKOS for ontologies presented in the Simple Knowledge Organization System (SKOS) language and ONKI-Geo for geographical ontologies with a map interface. The presented ONKI systems are operational on the web, used in the National Finnish Ontology Service. They have been successfully used in several pilot applications.

1 Ontologies as Web 2.0 Services

The Semantic Web[1] introduces a metadata layer on top of the World Wide Web infrastructure for describing its content and services in an explicit, machine "understandable" way using ontologies [1–3]. When such content is available, semantically aware applications for e.g. searching and browsing the distributed content can be created, as demonstrated in e.g. various semantic portals [4–6]. Many ontologies have been created and are available online in RDF(S) and OWL form today. For example, the Swoogle[2] [7] search engine index contains over 10,000 ontologies on the web.

One of the main lessons learned in our work on creating semantic portals [5, 8, 9, 6] is that metadata in data sources, such as museum databases, are often syntactically heterogeneous and contain spelling errors, are semantically ambiguous, and are based on different vocabularies [10]. This results in lots of tedious syntactic correction, semantic disambiguation, and ontology mapping work when making the contents semantically interoperable, and when publishing them on

[1] http://www.w3.org/2001/sw/
[2] http://swoogle.umbc.edu/

the Semantic Web. A natural solution to this problem would be to enhance legacy cataloguing and content management systems (CMS) with ontological annotation functions so that the quality of the original data could be improved and errors fixed in the content creation phase. However, implementing such ontological functions in existing legacy systems may require lots of work and thus be expensive, which creates a severe practical hindrance for the proliferation of the Semantic Web.

This relates to the more general challenge of the Semantic Web today: ontologies are typically published as files without support for using them in applications. Each application tends to re-implement similar functions for utilizing ontologies, such as semantic autocompletion and disambiguation [11], browsing and finding concepts, and populating ontologies. It is like re-creating map services from scratch in different geographical web applications, rather than using available services such as Google Maps[3], Yahoo Maps[4], or Microsoft Live Search Maps[5]. We argue that ontologies should be published as lightweight shared services which can be easily utilized in legacy systems using a mash-up approach in the same spirit as e.g. Google Maps, Yahoo Maps and Freebase[6] are used today. This approach for publishing ontologies means, that generic, shared functionalities are combined with specific applications using lightweight scripting and programming technologies such as Ajax[7].

In the following, we first outline the requirements of mash-up ontology services and present the implementation of a generic mash-up ONKI Ontology Service and framework which is currently used in the National Ontology Service in Finland[8]. We then present two implementations of ontology specific server implementations conforming to the ONKI Service framework: ONKI-SKOS for general SKOS[9] ontologies and ONKI-Geo [12] for geographical ontologies. After this, utilization of ONKI services in an external application is discussed by presenting two application scenarios. Finally, contributions, results, and lessons learned are summarized, and directions for further research outlined.

2 Requirements of a Mash-Up Ontology Service

The national semantic web infrastructure model being built by the FinnONTO project in Finland [13] argues that ontology services are needed for three major user groups: 1) *Ontology developers* need a collaborative ontology development, versioning, and publishing environment for ontologies [14]. 2) *Content indexers* need services for finding the desired annotation concepts and for transporting the corresponding URIs and other data from the ontology service into external

[3] http://maps.google.com/
[4] http://maps.yahoo.com/
[5] http://maps.live.com
[6] http://code.google.com/p/freebase-suggest/
[7] http://en.wikipedia.org/wiki/Ajax_(programming)
[8] http://www.yso.fi
[9] http://www.w3.org/2004/02/skos/core/

applications. 3) *Information searchers* need services for finding and disambiguating keyword meanings, and for transporting the corresponding URIs into search engines and other applications. In this paper we focus on the problem of supporting content indexing in applications and legacy systems.

Ontology servers are intended for managing ontologies, providing support for designing, choosing and accessing ontologies [15–17]. However, compared to previous work on ontology servers, we propose the idea of creating ontology services which can easily be used in applications. This requires the following features:

Mash-up integration support. Ontology servers should support runtime integration of the functionalities to applications and legacy systems, especially for annotation and semantic search.

Semantic autocompletion and disambiguation. Efficient search functionalities are important when trying to find the semantically correct concepts from large ontologies. Text search boosted up with semantic autocompletion and disambiguation functionalities [11] supports the user in finding the right concept by giving constant feedback of the query, and by helping in disambiguating the intended concept meaning.

Concept fetching. When using ontologies in combination with other applications, the idea of "copying" or "transferring" concepts between applications is important. We propose a concept fetching functionality for moving concept URIs from the ontology server to the target application, such as a legacy cataloguing system or CMS. To support legacy systems, indexing terms (concept labels) should be possible to use instead of URIs even though this may create disambiguation and mapping problems e.g. between ontology versions due to potentially less specific identifiers than the URIs.

Concept collecting. Usually no single concept describes all the aspects of the entity that is being described with a certain metadata property such as *dc:subject*. Therefore, it should be possible to collect multiple concepts from the ontology server and return these as a combination value in a specific metadata field to the legacy system.

Domain-specific user interfaces. The concepts of an ontology are typically visualized as an abstract graphical tree or graph visualization of the currently selected concept with its semantic vicinity [18]. Complementing this, we propose providing domain-specific interfaces, such as a map interface for geographical ontologies, when applicable.

3 ONKI Ontology Service

The ONKI Ontology Service is a general ontology library and framework that provides functionalities for accessing the ontologies using ready-to-use mash-up web widgets as well as application interfaces for humans and machines for doing, e.g., content indexing, concept disambiguation, searching and fetching. The service is based on ontology and domain specific implementations of ONKI Servers which conform to the ONKI interfaces. This means that it is possible to

provide a single mash-up web widget to access all ontologies but at the same time provide domain-specific user interfaces and technical implementations optimized for ontologies of different sizes, modelling languages, etc.

The ONKI Widget (Figure 1) is a ready-made user interface widget for using the ONKI Service in content annotation (indexing). It enables the user, e.g. a content annotator, to find the correct ontological concepts and their URIs and then transfer the URIs and the concept labels to their own content management application. Such a simple means for getting the URIs and to use them in applications is crucial for enabling the content creation on the Semantic Web.

In the following, the JavaScript and Direct Web Remoting (DWR)[10] based implementation of the widget is described which is intended to be used for extending HTML forms with ontology functionalities. However, the proposed solution is more general because in the case of other user interface technologies such as Java Swing, the ONKI Web Service[11] interface could be used to implement user interface technology specific implementations of the ONKI Widget.

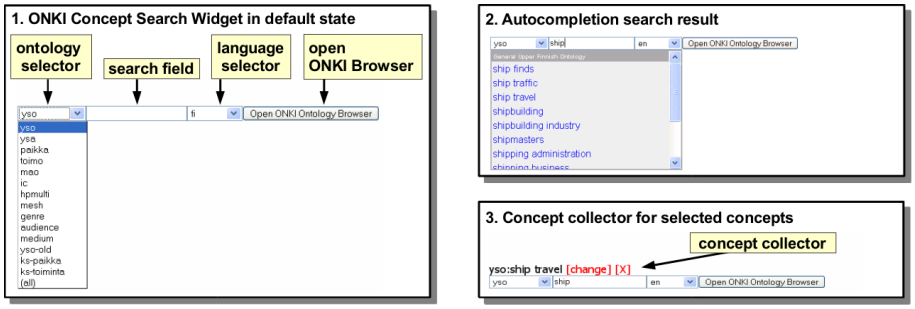

Fig. 1. ONKI Concept Search Widget.

Part 1 of Figure 1 shows the default components of the widget. The ontology selector can be used to change the ontology used in search or to select all ontologies as target for the search. The search field is used for finding concepts using text queries. In part 2 of Figure 1 the user is typing a search string to the autocompletion search field which dynamically performs a query after each input character (here "s-h-i-p-...") to the ONKI service and returns the concepts whose labels match the string, given the language selection. The results of the query are shown in the web widget's result list below the input field. In the case of synonym terms, the preferred label of a concept will be presented. For example, when searching for an (outdated) term "birch sugar", the system returns "birch sugar → xylitol" which means that "xylitol" is the preferred term.

When a desired concept is selected from the result list, it's URI and label are

[10] http://getahead.org/dwr
[11] http://www.w3.org/TR/ws-arch/

put in the widget's concept collector (Part 3 in Figure 1) for further usage such as submitting the content to the server application or accessing the collector from an application specific JavaScript program. The idea of the concept collector can be compared to the idea of shopping carts in web stores. In the example, the concept "ship travel" has been put into the concept collector.

The language of concept labels used in matching the query string can be chosen by using the language selector. The choice of languages depends on the ontology selected. For example, for YSO, English and Swedish are supported in addition to Finnish, and the Finnish Geo-ontology[12] can be used in Finnish, Swedish, and in three dialects of Sami spoken in Lapland. It is possible to use all languages simultaneously.

If the user doesn't know what to type in the text search field, the alternative of using a browsing interface is available by using the domain specific ONKI Browser ("Open ONKI Ontology Browser" button). The ONKI Browser can also be used for disambiguating homonym terms, i.e. concepts with identical labels, by aiding the user to inspect the context of the concepts. When the desired concept has been found using the ONKI Browser, the concept's URI and label are fetched into the application by pressing the "Fetch concept" button on the ONKI Browser page corresponding to the concept. Two implementations of the ONKI Browser are presented in the next section of the paper.

The web widget can be integrated into an HTML Form with two lines of JavaScript code. The following code line loads the ONKI JavaScript library and should be added into the HEAD section of the HTML page:

```
<script type="text/javascript"
        src="http://www.yso.fi/onki.js"></script>
```

Using the library, ordinary HTML form input fields can be extended with ONKI functionality by declaring the *onkeyup* event handler for the field. For example, adding the General Finnish Upper Ontology YSO to a example *dc:subject* field is done as follows:

```
<input id="dc:subject" onkeyup="onki['yso'].search()" />
```

As a result, when a page is accessed, the user interface is enhanced with ontology support. The widget provides a default concept collector (Part 3 in Figure 1) which shows the fetched concepts in the widget's user interface and stores them in hidden input fields. When the form is submitted, the values of the hidden input fields can be processed by the target application in the same way as any HTML form submissions.

The ONKI Widget can be customized by configurations and by implementing callback functions. Configuration possibilities include disabling the menus for selecting ontologies and the language, the search field, or the "Open ONKI Browser" button. The widget can also be configured to restrict the search to concepts of certain type or belonging to a specific subtree of an ontology. Additionally, CSS styling can be used for configuring the appearance of the widget. The

[12] http://www.seco.tkk.fi/ontologies/suo/

ONKI Widget's callback functions enable application specific implementations of e.g. the concept collector or the concept search result list using JavaScript.

The ONKI API includes the following methods:

- *search(query, lang, maxHits, type, parent)* - for searching for ontological concepts. Returns a list of hits.
- *getLabel(URI, lang)* - for fetching a label for a given URI in a given language.
- *getAvailableLanguages()* - for querying for supported languages of an ontology. Returns a list of languages.

By implementing these methods, any system can be added to the ONKI Service to be used via the general ONKI Service functionalities such as the ONKI Widget. This is demonstrated by the case implementations presented below. Thus, the ONKI Service is not tied to a single ONKI Server implementation.

4 Two domain specific ONKI Server implementations

Two domain-specific ONKI Servers have been implemented conforming to the general ONKI service functionalities described in the previous section. ONKI-SKOS is intended for lightweight ontologies and ONKI-Geo [12] for geographical ontologies. In the following these two systems are shortly described.

4.1 ONKI-SKOS Server for SKOS Vocabularies

ONKI-SKOS is a general ontology service supporting thesaurus-like ontologies especially in content indexing. ONKI-SKOS can be used to browse, search and visualize any vocabulary conforming to the SKOS recommendation, and also RDF(S) and OWL ontologies with additional configuration. ONKI-SKOS does simple reasoning, e.g. transitive closure over class and part-of hierarchies. The implementation has been tested using various ontologies, e.g. the General Finnish Upper Ontology YSO, containing 20,000 concepts.

ONKI-SKOS Browser (Figure 2) is the graphical user interface of ONKI-SKOS. It consists of three main components: 1) *concept search with semantic autocompletion*, 2) *concept hierarchy* and 3) *concept properties*. When typing text to the search field, a query is performed to match the concepts' labels. The result list shows the matching concepts, which can be selected for further examination. The search can be further narrowed by restricting the search to concepts of a certain type or to a desired subtree of the ontology.

When a concept is selected in ONKI-SKOS Browser, its concept hierarchy is visualized as a tree structure, and its properties are shown as a table. Various configuration properties are specified to enable ONKI-SKOS to process the ontologies as desired. The configurable properties include the ontological properties used in concept hierarchy generation, the properties used to label the concepts, the concept to be shown in the default view and the default concept type used in restricting the searches.

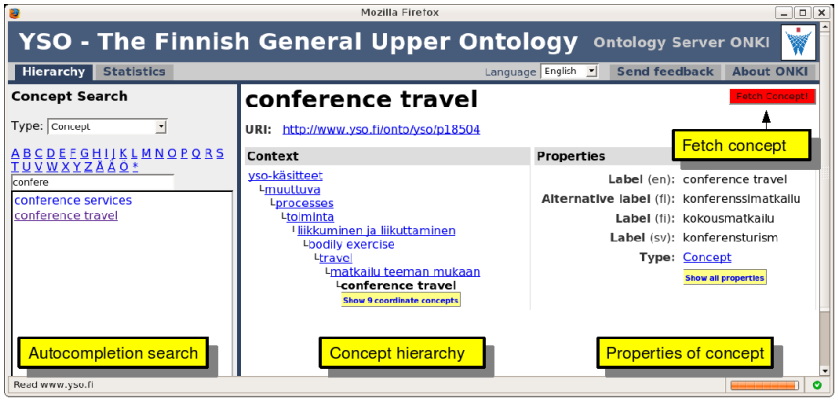

Fig. 2. ONKI-SKOS Browser.

ONKI-SKOS Server is implemented as a Java Servlet using the Jena Semantic Web Framework[13], the DWR library and the Lucene[14] text search engine.

4.2 ONKI-Geo Server for Geographical Ontologies

ONKI-Geo [12] is an ontology service specialized for geographical data. It was developed for the Finnish Place Ontology SUO (Suomalainen Paikkaontologia) [19] which currently has been populated with 1) place information from the Geographic Names Register (GNR) provided by the National Land Survey of Finland[15] and with 2) place information from the GEOnet Names Server (GNS)[16] maintained by the National Geospatial-Intelligence Agency (NGA) and the U.S. Board on Geographic Names (US BGN). GNR contains about 800,000 multilingual resources of natural and man-made features in Finland, including data such as place type or feature type and the coordinates of a place. The GNS register contains similar information about 4,100,000 places around the world.

The ONKI-Geo Browser (Figure 3) is intended for annotating resources using unambiguous place identifiers (URIs) or coordinates for arbitary points or polygons. Using unambiguous place identifiers is useful e.g. due to homonymous place names: there are hundreds of places in Finland with the name “Isosaari” (“Big Island”). The ONKI-Geo Browser contains several facets for narrowing the search to find the intended place instance: A polygon can be drawn in the map interface for making a search on all places in the selected area. The other facets are an ontology of geographic features (e.g., lake, city, etc.), the languages of the place names, and a place name search with autocompletion. The system uses Google Maps widgets for visualizating the places.

[13] http://jena.sourceforge.net/
[14] http://lucene.apache.org/java
[15] http://www.maanmittauslaitos.fi/
[16] http://earth-info.nga.mil/gns/html/

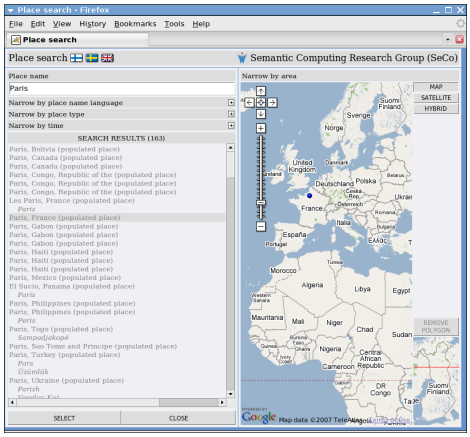

Fig. 3. ONKI-Geo Browser. Search can be constrained by using the facets on the left or by drawing a polygon on the map. By pushing the "Select" button in the left bottom corner, the concept or selected coordinate information is transferred into the mash-up widget.

5 Integrating ONKI with Application Systems

In the following we describe use cases of the ONKI system.

5.1 Integrating ONKI with a Cataloguing System

To demonstrate how to add ONKI functionalities to a legacy system, we created a simple web form (part 1 of Figure 4) presenting the MuseumFinland [5] metadata fields [20]. By adding the ONKI Concept Search Widget to the fields, ontologies can be used in annotating museum collection items. Part 2 of Figure 4 depicts the original form after adding the widgets.

After this the form can be used for creating semantic metadata. If the underlying system does not support URIs, the system has to be modified to handle this kind of information. Alternatively the ONKI Widget can be configured to return concept labels instead of URIs.

5.2 An Annotation Editor Based on ONKI Ontology Services

SAHA[17] [21, 22] is a generic annotation system supporting distributed collaboration in creating annotations, and hiding the complexity of the annotation schema and the domain ontologies from the annotators. SAHA adapts to different metadata schemas, which makes it suitable for different applications. Support for using ontologies is based on ONKI ontology services (Figure 5). The system

[17] http://www.seco.tkk.fi/services/saha/

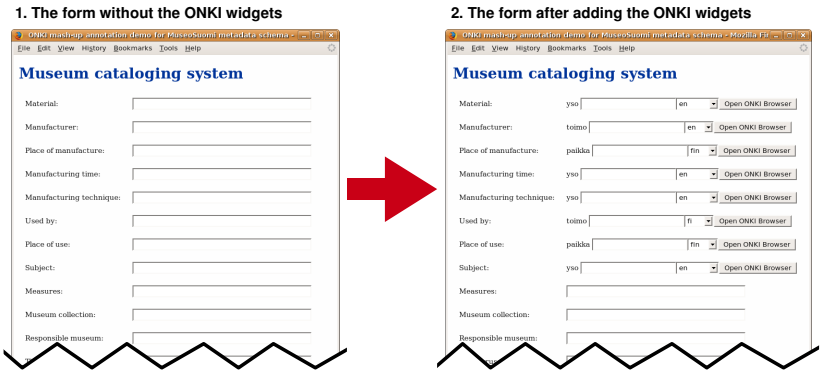

Fig. 4. A museum cataloguing system before and after integrating the ONKI widgets.

is being tested in various practical semantic portal projects such as HealthFinland [6] and CultureSampo [23].

The metadata field elements are implemented using the ONKI widget, as discussed above. Depending on the field, different ONKI servers are used as specified in the SAHA configuration. In this case the Finnish General Upper Ontology YSO [24], published as an ONKI service[18], is used for selecting annotation concepts. SAHA can also make use of the automatic text extraction component POKA[19] in extracting potential annotation concepts from web resources [21], and populating the SAHA concept collector with them.

Annotations created with SAHA are stored in a centralized database, from which they can be retrieved for editing or to be used in applications such as semantic portals. It is possible to view and edit existing annotations by reading the metadata fields in corresponding widget collectors. Furthermore, SAHA supports population of its own annotation ontologies by new resources. In this way, different users creating annotations collaboratively can share new resources created by anyone, e.g., instances of new works of art or other artifacts.

6 Discussion

The main contribution of this paper is to present the idea of publishing *ontologies as mash-up services* that can be integrated in a lightweight fashion to legacy systems on the user interface level. To demonstrate the applicability of the idea, we presented the ONKI service and two implementations of the ONKI interface: the general ontology server ONKI-SKOS and the geographical ontology server ONKI-Geo. The two ONKI implementations also demonstrated the idea of creating domain specific user interfaces to better support the usage of different

[18] http://www.yso.fi/onto/yso/
[19] http://www.seco.tkk.fi/tools/poka/

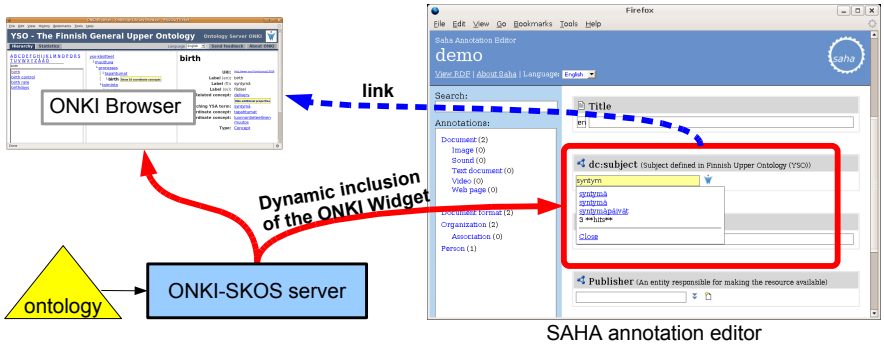

Fig. 5. ONKI integrated with the SAHA annotation editor.

types of ontologies. A practical contribution of the paper was to introduce the idea of concept fetching between applications and the need for concept collecting when using an ontology server for annotation purposes. Finally, semantic autocompletion was proposed and implemented in the user interface components to provide an efficient method for finding and disambiguating concepts.

A lesson learned from implementing the concept fetching functionality as a web browser application was that special tricks are needed to transfer data between browser windows loaded from different domains[20]. Security being an important concern, we suggest that browsers should provide some standardized solution for communication between domains.

The widget is currently being developed for supporting other tasks where ontological concepts need to be searched and fetched, such as ontological content search. Future work includes researching how the autocompletion concept search could help even more in disambiguation the concepts without forcing the user to open the ONKI Browser with all information about the concepts. In future, other user interface environments than HTML and the web browser could be supported, such as Java Swing. The ONKI API does not currently use RDF for returning the content to the Widget to make it easier to handle the content with JavaScript. However, in future RDF might be used. Finally, according to our vision of a national ontology service, the ontologies in such a service should be extensively and mutually interlinked to support creating cross-domain applications. Therefore, the question of how to support developing and using mutually interlinked ontologies should be researched.

[20] Since the ONKI-Browser is located in a different domain than the ONKI-Widget, the communication between them was solved as follows: the Widget opens a new browser window, which contains the ONKI-Browser in an IFRAME. The selected concept URI is returned from the ONKI-Browser by changing the fragment identifier of the window with the IFRAME, which can be accessed by the Widget.

Acknowledgments

We thank Ville Komulainen for his work on the first version of the ONKI server, and Robin Lindroos and Tomi Kauppinen for collaboration in ONKI-Geo development. Our research is a part of the National Semantic Web Ontology Project in Finland[21] (FinnONTO) 2003–2007 funded by the Finnish Funding Agency for Technology and Innovation (Tekes) and 36 companies and public organizations.

References

1. Gruber, T.R.: A translation approach to portable ontology specification. Knowledge Acquisition **5**(2) (June 1993) 199–220
2. Staab, S., (eds.), R.S.: Handbook on ontologies. Springer-Verlag (2004)
3. Fensel, D.: Ontologies: Silver bullet for knowledge management and electronic commerce (2nd Edition). Springer-Verlag (2004)
4. Reynolds, D., Shabajee, P., Cayzer, S.: Semantic Information Portals. In: Proceedings of the 13th International World Wide Web Conference on Alternate track papers & posters, New York, NY, USA, ACM Press (May 2004)
5. Hyvönen, E., Mäkelä, E., Salminen, M., Valo, A., Viljanen, K., Saarela, S., Junnila, M., Kettula, S.: Museumfinland—Finnish museums on the semantic web. Journal of Web Semantics **3**(2) (2005) 25
6. Hyvönen, E., Viljanen, K., Suominen, O.: Healthfinland—Finnish health information on the semantic web. In: Proceedings of the 6th International Semantic Web Conference (ISWC 2007), Busan, Korea, Springer-Verlag (Nov 2007)
7. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V., Sachs, J.: Swoogle: a search and metadata engine for the semantic web. In: CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management, New York, NY, USA, ACM Press (2004) 652–659
8. Sidoroff, T., Hyvönen, E.: Semantic e-goverment portals - a case study. In: Proceedings of the ISWC-2005 Workshop Semantic Web Case Studies and Best Practices for eBusiness SWCASE05. (Nov 2005)
9. Känsälä, T., Hyvönen, E.: A semantic view-based portal utilizing Learning Object Metadata (August 2006) 1st Asian Semantic Web Conference (ASWC2006), Semantic Web Applications and Tools Workshop.
10. Hyvönen, E., Salminen, M., Kettula, S., Junnila, M.: A content creation process for the Semantic Web (2004) Proceeding of OntoLex 2004: Ontologies and Lexical Resources in Distributed Environments, May 29, Lisbon, Portugal.
11. Hyvönen, E., Mäkelä, E.: Semantic autocompletion. In: Proceedings of the first Asian Semantic Web Conference (ASWC 2006), Beijing, Springer-Verlag, New York (August 4–9 2006)
12. Kauppinen, T., Henriksson, R., Sinkkilä, R., Lindroos, R., Väätäinen, J., Hyvönen, E.: Ontology-based disambiguation of spatiotemporal locations. In: 1st international workshop on Identity and Reference on the Semantic Web (IRSW2008), 5th European Semantic Web Conference 2008 (ESWC 2008), Tenerife, Spain. (June 1-5 2008) forthcoming.

[21] http://www.seco.tkk.fi/projects/finnonto/

13. Hyvönen, E., Viljanen, K., Tuominen, J., Seppälä, K.: Building a national semantic web ontology and ontology service infrastructure—the finnonto approach. In: Proceedings of the European Semantic Web Conference ESWC 2008, Springer (June 1-5 2008)
14. Komulainen, V., Valo, A., Hyvönen, E.: A tool for collaborative ontology development for the semantic web. In: Proc. of the International Conference on Dublin Core and Metadata Applications (DC 2005). (Nov 2005)
15. Ahmad, M.N., Colomb, R.M.: Managing ontologies: a comparative study of ontology servers. In: ADC '07: Proceedings of the eighteenth conference on Australasian database, Darlinghurst, Australia, Australia, Australian Computer Society, Inc. (2007) 13–22
16. Ding, Y., Fensel, D.: Ontology library systems: The key to successful ontology reuse. In: Proceedings of SWWS'01, The first Semantic Web Working Symposium, Stanford University, USA. (2001) 93–112
17. Komulainen, V.: Public services for ontology library systems. Master's thesis, University of Helsinki, Department of Computer Science (January 2007)
18. Eklund, P., Roberts, N., Green, S.: Ontorama: Browsing rdf ontologies using a hyperbolic-style browser. In: First International Symposium on Cyber Worlds, CW02, Theory and Practices, IEEE Press. (2002) 405–411
19. Kauppinen, T., Henriksson, R., Väätäinen, J., Deichstetter, C., Hyvönen, E.: Ontology-based modeling and visualization of cultural spatio-temporal knowledge. In: Developments in Artificial Intelligence and the Semantic Web. Proceedings of the 12th Finnish AI Conference STeP 2006. (October 26–27 2006)
20. Viljanen, K., Tuominen, J., Känsälä, T., Hyvönen, E.: Distributed semantic content creation and publication for cultural heritage legacy systems. In: Proceedings of 2008 IEEE International Conference on Distibuted Human-Machine Systems, IEEE Press (2008)
21. Valkeapää, O., Alm, O., Hyvönen, E.: Efficient content creation on the semantic web using metadata schemas with domain ontology services (system description). In: Proceedings of the European Semantic Web Conference ESWC 2007, Innsbruck, Austria, Springer (June 4–5 2007)
22. Valkeapää, O., Hyvönen, E.: A browser-based tool for collaborative distributed annotation for the semantic web. In: Proceedings of the Semantic Authoring and Annotation Workshop, 5th International Semantic Web Conference. (November 2006)
23. Hyvönen, E., Ruotsalo, T., Häggström, T., Salminen, M., Junnila, M., Virkkilä, M., Haaramo, M., Mäkelä, E., Kauppinen, T., , Viljanen, K.: Culturesampo–finnish culture on the semantic web: The vision and first results. In Robering, K., ed.: Information Technology for the Virtual Museum. LIT Verlag, Berlin. (Nov 2007)
24. Hyvönen, E., Viljanen, K., Mäkelä, E., Kauppinen, T., Ruotsalo, T., Valkeapää, O., Seppälä, K., Suominen, O., Alm, O., Lindroos, R., Känsälä, T., Henriksson, R., Frosterus, M., Tuominen, J., Sinkkilä, R., Kurki, J.: Elements of a national semantic web infrastructure—case study finland on the semantic web (invited paper). In: Proceedings of the First International Semantic Computing Conference (IEEE ICSC 2007), Irvine, California. (September 2007) IEEE Press.