ODESeW. Automatic Generation of Knowledge Portals for Intranets and Extranets

O. Corcho, A. Gómez-Pérez, A. López-Cima,
V. López-García, M.C. Suárez-Figueroa

Laboratorio de Inteligencia Artificial
Facultad de Informática
Universidad Politécnica de Madrid
Campus de Montegancedo sn.
Boadilla del Monte, 28660. Madrid, Spain
{ocorcho, asun}@fi.upm.es, {alopez, vlopez, mcsuarez}@delicias.dia.fi.upm.es

Abstract: This paper presents ODESeW (Semantic Web Portal based on WebODE platform [1]) as an ontology-based application that automatically generates and manages a knowledge portal for Intranets and Extranets. ODESeW is designed on the top of WebODE ontology engineering platform. This paper shows the service architecture that allows configuring the visualization of ontology-based information for different kinds of users, establishing reading and updating access policies to its content, and performing consistency checking between the portal information and the ontologies underlying it.

1. Introduction

The terms *knowledge portal*, *semantic portal* and *community web portal* can be found in the literature ([6][10]) to indistinctly refer to knowledge-based web sites that allow corporate access to information and applications. A good definition of what they are can be found in [6], where they are defined as web applications that "*provide the means to select, classify and access, in a semantically meaningful and ubiquitous way, various information resources (e.g., sites, documents, data) for diverse target audiences (corporate, inter-enterprise, e-marketplace, etc.).*" From now on, we will use the term "knowledge portal" to refer to this kind of applications.

Knowledge portals present structured views of the web according to what it is usually called a knowledge catalogue [6]. A knowledge catalogue holds descriptions about the resources available to the community members, and is more flexible and complex than conventional (relational or object) databases. Ontologies are commonly used for this task of structuring knowledge, since they represent shared knowledge within a community.

The process of content provision in knowledge portals is usually performed collaboratively, normally with few resources (manpower, money) [10]. This supposes a great effort to maintain the Web portal and to integrate the information it contains

(even if it is using ontologies to structure it). Besides, content presentation is always a hard task, especially in knowledge-intensive web sites where content is continuously updated. To ameliorate the hard task of knowledge portal management, we need applications that automate these difficult knowledge workflow processes (content provision and integration, content presentation, and content access), as well as frameworks that support them.

Furthermore, in this knowledge intensive portals we distinguish between ontology developers and content (knowledge asset) providers versus Intranet and Extranet users. Ontology developers are in charged of developing the ontologies, which will be used by the content providers as a primary piece of knowledge for describing knowledge assets and by the end users as an index used to browse the knowledge portal. Regarding the end users, we distinguish between Intranet and Extranet users. Intranet users, which are also content providers, access content inserted by themselves or by others members. Different Intranet users have different permissions either for inserting content on the knowledge portal or for browsing the collected assets. Finally, extranet users, who scarcely include new content, but mainly access the allowed content by the knowledge portal administrator. Therefore, knowledge portals must be created having in mind that they have to act both as Intranets (private networks contained within an enterprise, whose main purpose is to share company information and computing resources among employees) and as Extranets (defined as collaborative extensions of an Intranet, which expands access to individuals outside the company) [1].

In this paper, we present ODESeW, an ontology-based application built inside the WebODE ontology engineering workbench, that allows managing knowledge-intensive ontology-based Intranets and Extranets, providing the following functions:

- *Knowledge modelling,* by means of an ontology development platform that integrates several ontology development services. As the knowledge portal will be used on the web, is highly recommend to use an ontology server (and not a stand-alone ontology editor) that allows to build cooperatively the ontologies as well as to access the ontologies through the web. The use of an ontology server as a basic infrastructure over which the knowledge portal is built will ease the management in a sync way of the assets with respect to the ontology changes. From a software perspective, the knowledge portal will benefit from the present and further services provided by the ontology server.
- *Content editing/provision* by means of ontology instances editing. ODESeW allows inserting, updating and removing class instances, their attributes and relation instances, in multiple interlinked ontologies and with different editing permissions for the portal users. As part of the instance editing functions, ODESeW can be also used as a document management tool, which allows handling electronic documents.
- *Content presentation/visualization* by means of highly-configurable user-defined visualizations of ontology classes, relations and instances, and with different browsing permissions for the portal users. The ontology is used for indexing the knowledge assets and for browsing them accordingly. The ontology provides structure on this content that helps the user find knowledge assets. The content

[1] Both definitions are obtained from http://www.scotsmist.co.uk/glossary_e.html

stored in the portal can be accessed dynamically with menus automatically generated from ontologies according to the user's permission, visualizing differently the different types of information stored in the knowledge assets; an example is the shallow natural language generation functions for Extranet users. The knowledge portal also provides annotated markup of its assets in RDF(S), DAML+OIL and OWL.

- *Content search and querying* functions, based on a hybrid approach based on ontologies and keywords. The content search and querying modules use the WebODE API for accessing and querying the contents of the ontologies.
- Easy web site *administration* services, which allow managing the knowledge portal users, editing and visualization permissions, and several other portal management needs. Such services are only accessed by the users belonging to the knowledge portal administration group.

As an important advantage of ODESeW over other similar Knowledge portals (we must cite the ontoweb portal) is the automatic synchronization between the contents of the portal and the ontologies in which it is based. So, if an ontology is modified with the WebODE ontology editor, the changes will be automatically seen in the knowledge portal either for ontology conceptualization itself or for its instances.

The paper is structured as follows: section 2 presents the software architecture of ODESeW, paying special attention to its integration with the WebODE ontology engineering workbench. Section 3 describes the most relevant functions of ODESeW, grouped in content editing, content presentation, content search and querying, and portal administration. Section 4 shows a case study of the use of ODESeW in a real application: the Intranet and Extranet of the European funded project Esperonto. In section 5, we describe some related work, and we conclude and present further work in sections 6.

2. ODESeW Architecture

ODESeW has been built in the framework of WebODE, a scalable ontology engineering workbench that gives support to the ontology building methodology METHONTOLOGY [5].

As shown in figure 1, the ODESeW portal is one of the two main front-end applications of the WebODE workbench. The other one is the WebODE ontology editor, which integrates all the ontology editing and management functions of the platform.

WebODE is platform-independent, since it is completely implemented in Java. To allow scalability and easy extensibility, it is supported by an application server, so that services can be easily created and integrated in the workbench by means of a management console. One important advantage of using this application server is that it allows deciding which users or user groups may access each of the services of the workbench.

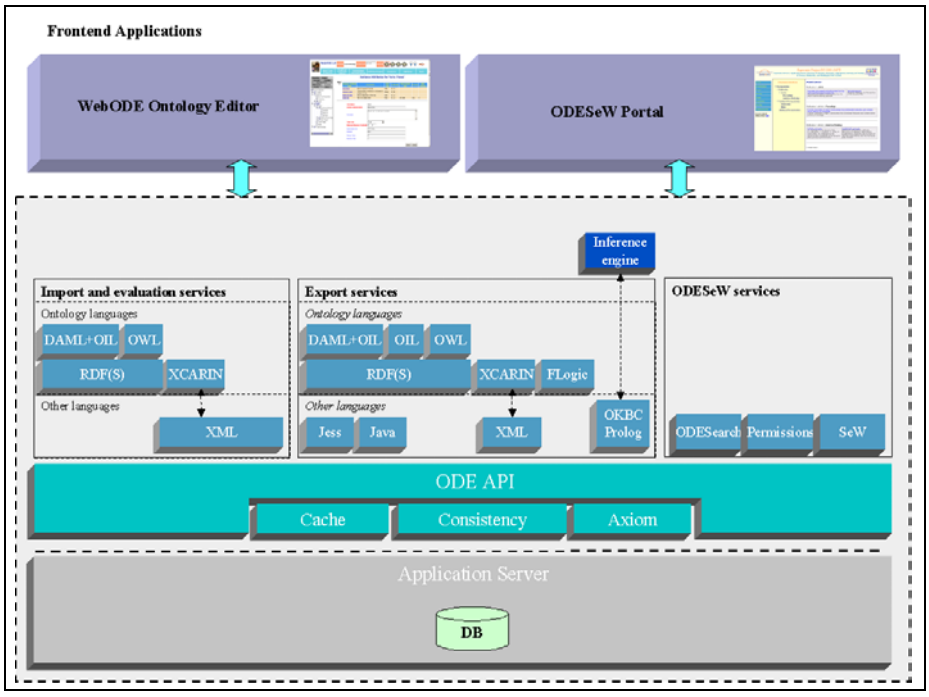

Figure 1. WebODE ontology engineering workbench architecture.

The figure also shows the most relevant services currently available in the WebODE workbench. The core of the WebODE's ontology development services are: the cache, consistency and axiom services, and the ontology access service (ODE API), which defines an API for accessing WebODE ontologies. One of the main advantages of this architecture is that these services can be accessed remotely from any other application or any other instance of the WebODE workbench.

Furthermore, ontologies are stored in a relational database, so they can manage huge ontologies quite efficiently. And it is also easily extensible, so that the database manager can be changed, or any backend system can be plugged in the bottom of the architecture. Finally, WebODE also provides backup management functions for the ontologies stored in the server.

The figure shows that the import, export and evaluation services are running on top of the ontology access service. These services import ontologies from XML, XCARIN, RDF(S)[3][7], DAML+OIL [11], and OWL [4]; and export ontologies to XML, FLogic, XCARIN, RDF(S), OIL, DAML+OIL, and OWL. Ontologies are also exported to languages that are not specifically created for defining ontologies, such as Prolog, Jess, and Java. For instance, the Prolog export service is used as a basis of the WebODE's inference engine. WebODE also evaluates ontologies written in RDF(S), DAML+OIL, and OWL.

Once described the main characteristics of the WebODE workbench, we will proceed to describe the services used by the ODESeW application. To implement

ODESeW, we have built three more services on top of the ODE API, as shown in the right of the figure: *ODESearch*, *permission* and *SeW*.

- *ODESearch* allows querying the WebODE ontologies, by means of keywords or using the attributes of the ontology concepts as templates, as will be explained in section 3.3.
- The *permission* service is in charge of managing security in the access to the concepts, instances and attributes of the ontologies. It will manage both read and write access permissions to the content stored.
- *SeW* gives support to the administration functions of the ODESeW application. It allows selecting which ontologies will be published in the portal, which types of users can access it (administrators, guest users, etc.), how instances in the ontology will be visualized in the portal, etc. These functions are described in section 3.4.

There are many advantages of having built ODESeW on top of the WebODE workbench. First of all, ODESeW can use any of the WebODE workbench services. For example, with the ontology import services we can import other ontologies in the workbench, and these new ontologies can be easily selected for publication in the ODESeW portal. Consequently, we can create a complete new knowledge portal (including its Intranet and its Extranet) in a very short period of time.

Another advantage is that we can edit any of the ontologies published with ODESeW using the WebODE ontology editor, and observe at run-time the modifications in the knowledge portal, which means that there is auto-synching of the portal with respect to the ontology.

3. ODESeW functions

ODESeW generates automatically knowledge portals for Intranets and Extranets, both of which use the same assets and knowledge. The knowledge portal provides different functions in each case:

- If the knowledge portal is being used as an Intranet, corporate users will be able to insert and update content in the portal as content providers, browse the content that they have inserted or that other corporate members have inserted there, and perform searches and queries on that content. The ontologies issues either for indexing knowledge asset or for searching them more efficiently.
- If the knowledge portal is being used as an Extranet, external users will usually be able to edit very restricted parts of the content stored in the portal, and browse, query and search only the content identified as public content by the content providers.

Apart from these content provision, visualization, and access functions, ODESeW provides management services that allow configuring them.

In this section, we will present the main functionalities of ODESeW, grouped in the four categories of: content editing/provision, content presentation/visualization, content querying and search, and administration services.

Another interesting function in a knowledge portal is the possibility to modify the published ontologies. ODESeW does not give support to this function, since for this task the WebODE ontology editor can be used. All the changes done to ontologies with the WebODE ontology editor are viewed in run-time execution in ODESeW, with no need to restart the web server.

3.1 Content editing/provision

In an ontology-based knowledge portal, the provision of content mainly consists of editing concept instances, that is, inserting, updating and removing instances of ontology concepts. ODESeW gives support to this task by allowing users to edit concept instances and the values of their attributes, and to connect such instances by means of relations, even if they belong to different ontologies. The ontology conceptualization editing is delegating to the WebODE application.

This content provision task in ODESeW is mainly performed by Intranet users, although ODESeW does not restrict it to Extranet users (the administrator of the knowledge portal may decide whether to give Extranet users editing permissions to specific parts of the ontologies published in the portal).

ODESeW gives support to the following content provision functions:

- *Instance creation and removal*. Users can create instances of any concept in any of the ontologies published in ODESeW, provided that the users have enough permissions to create such instances. The same occurs with the removal of concept instances.
- *Instance editing.* To edit instances, users are presented with the attributes of the concept from which the instance is instance of, as well as the attributes inherited through the concept taxonomy (multiple inheritance is allowed in ODESeW). Figure 2 shows the instance editing form for the instance *Angel López-Cima* of the concept *PhD Student*. The user can insert one or several values of any of these attributes, provided that their maximum cardinality and value type constraints are respected. All these constraints are checked by the WebODE platform.

  Depending on the attribute types, ODESeW gives different fields to insert and update their values. For instance, if the value type of an attribute is *Date*, ODESeW will present a calendar from where the user can select a specific date. In figure 2, the attribute *Date Of Birth* has a link to a calendar next to the Add Value button.

  If the value type is URL, the user can either insert directly a URL or upload a file (an image, a PDF document, etc.) that is converted to a URL inside the knowledge portal. In the example of figure 2, we could insert the URL *http://delicias.dia.fi.upm.es/~alopez* as a value of the attribute *Homepage*, and we have inserted an image file as a value for the attribute *Photo*.
- *Relation instance editing*. The bottom of the instance editing form shows the relations that can be applied to the instance being edited. In figure 2, we can see that a *PhD Student* can *belong to* an *Organization* and could be the *contact Person* of an *Organization*. These relations appear in the form because their domain is the concept *Person*, which is a superclass of *PhD Student*.

If a user decides to create any of these relation instances, (s)he will be shown a list of candidate instances from the same or from other ontologies that are instances of the concept that is the range of the relation. For instance, if we selected the relation *belong to* in figure 2, we would be shown a list of instances of the concept *Organization* or any of its subclasses. WebODE also checks the integrity constraints of these relations.

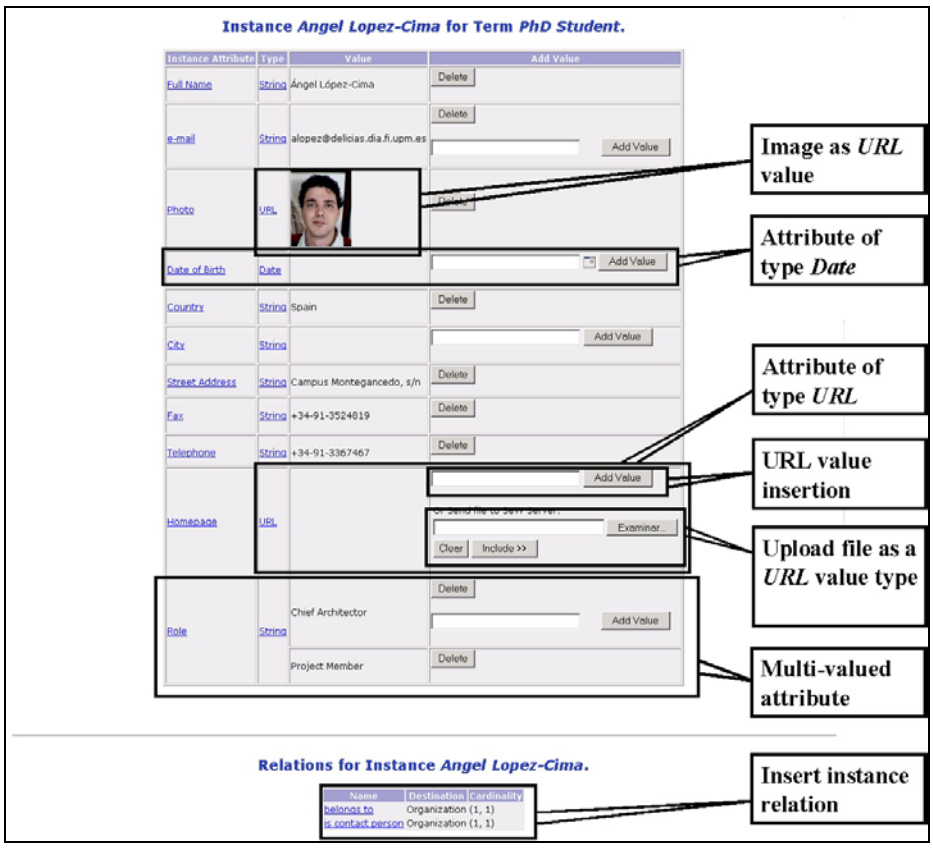

Figure 2. Editing form for the instance *Angel López-Cima* of the concept *Phd Student* in an ontology about *persons*.

3.2 Content presentation/visualization

Content visualization in an ontology-based knowledge portal mainly consists of showing the ontology concepts and their related instances, presenting the details of ontology instances and their relations with other instances, and allowing the navigation through these relations and between the different ontologies published in the portal. Although not related to the human-consumption, content visualization also consists of producing the annotated markup of all the knowledge stored in the portal.

It is important to mention again that the content is the same for Extranet and Intranet users, but ODESeW visualize differently depending on the two types of users.

On the one hand, if ODESeW is being used as an Extranet, that is, the user has not logged in the portal (from now on, we will refer to non-logged users as guest users), the portal "hides" all the knowledge representation terminology (words such as 'ontology', 'concept', 'instance', etc). So, external users do not need to know that the knowledge portal is internally based on ontologies. In fact, they do not need to know the terminology used in the knowledge representation field to use the portal. To present the content in a mode user-friendly way, ODESeW includes a shallow natural language generation functions.

On the other hand, if ODESeW is being used as an Intranet, that is, the user has logged in the portal, the portal shows all the information that the user has access to, without hiding the knowledge representation terminology nor using natural language generation functions. Since different users may have different access and write permissions for the content in the portal, we can consider that there are as many different views of the information as type of users of the Intranet.

Figures 3 and 4 show different visualizations of the same instance (a deliverable in the European project Esperonto) at the Extranet and the Intranet, respectively. In figure 4, the Intranet user can see and access to the information of the attribute *On-line version*, but a guest user can not see the attribute as shows in figure 3 because this is a private attribute.

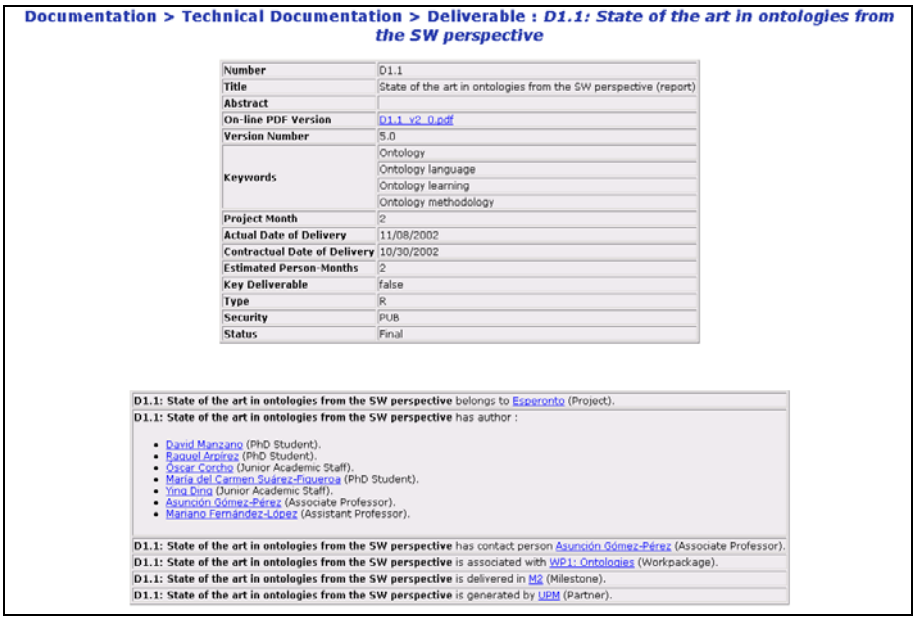

Figure 3. Instance detail for the Extranet.

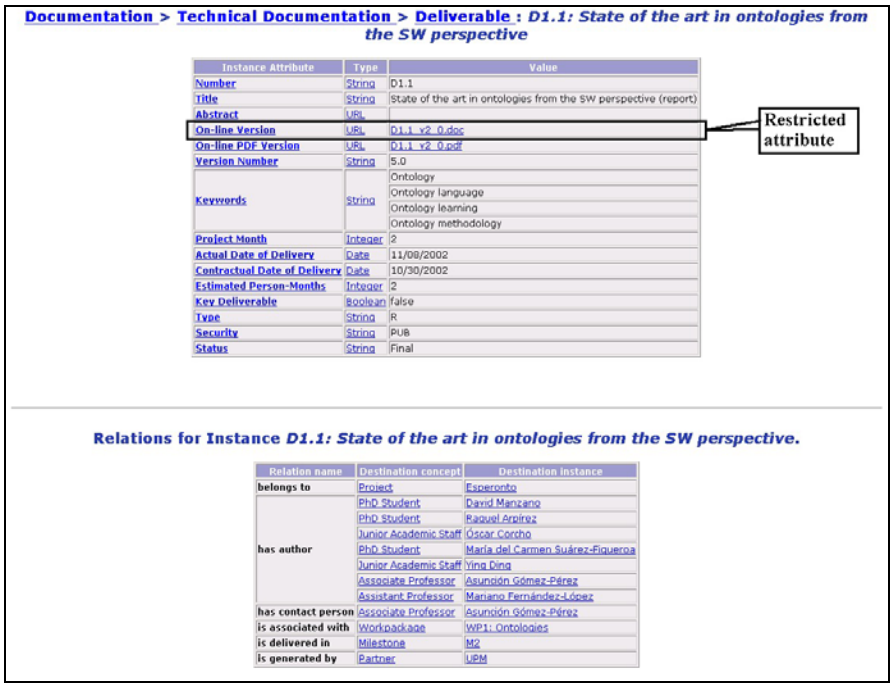

Figure 4. Instance view for Intranets.

Apart from the previous functions, ODESeW has the following content provision features:

- *Automatically generated menus* to access the ontologies published in the portal.
- *Automatic generation of concept taxonomies from ontologies* to browse each of the published ontologies according to the permissions defined for each user. For Extranet users, apart from taking into account read permissions, the concept taxonomy will not show concepts that have not instances avoiding to the users access to a concept without any information (instances).
- *Instance lists visualization*. The instances of a concept (direct instances) and all its subconcepts (indirect instances) are shown by selecting the concept in the previous concept taxonomies. Each of the listed instances may have a description based on one or more of its attributes (as described in section 3.4). The portal only shows accessible instances according to the user read permissions. Figure 5 shows an example of an instance list where the instances of the concept Organization are described by their full name and their logo.
- *Instance details visualization*. When visualizing an instance, users can see the attributes and relation instances for which they have read permissions.
  The visualization of each attribute is different depending on its value type (String, Date, URL, etc.). For instance, in figure 3 and 4, the URLs appear as links and the rest of attribute values appear as text boxes. URL attributes that contain an image file are visualized as images, as was shown in the editing form of figure 2.

Besides, the portal administrator can set the order in which the attributes of instances will be visualized, as we explain in section 3.4. By default, attributes are presented in alphabetical order.

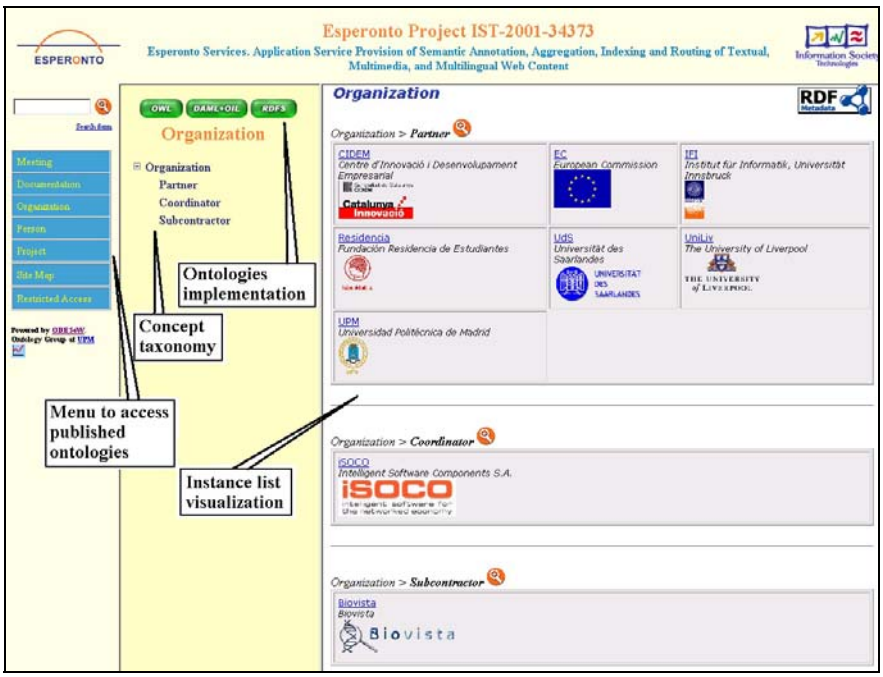

Figure 5. Instance list visualization.

- *Annotated ontology markup*. All the content visualized by the ODESeW is automatically generated as RDF code, and can refer to ontologies implemented in RDFS, DAML+OIL and OWL. To obtain this annotated markup, ODESeW uses the WebODE export services to these languages.

3.3 Content Search and Querying

In a data-intensive web-site, it is usually very difficult to find a specific piece of information, even in the case that content inside the portal is well structured. This problem is even more important in the case of the Extranet, since guest users do not need to have a clear idea of how the portal is structured.

For this reason, ODESeW includes a search engine that allows querying for information in one ontology or in all the ontologies of the portal. The search engine allows two kinds of search:

- *Keyword-based*. As in other conventional search tools, the search engine looks for instances or concept names that contain the keywords specified in the query.

- *Ontology-based*. ODESEW provides advanced search functions by means of a query form. The fields to be filled in at the query form are attributes and relations taken from the ontology. Once the user introduces the values (s)he is looking for, the search engines returns those instances that satisfy the conditions imposed in the attributes values specified in the form.

Though these are the most useful search and query facilities from the point of view of human-consumption, we must take into account that ODESeW also generates annotations in RDF for the content that it visualizes. This would allow other sites or agents to read the annotations and use that content in other environments.

3.4 Web site administration functions

ODESeW provides a set of management tools that can only be accessed by the administrator user. This tool suite gives support to the basic management functions needed to maintain the knowledge portal, namely user management, permission management (security), ontology publication management, attribute ordering and instance list descriptions.

- *User management*. With this function, the knowledge portal administrator can insert, remove or modify the users of the Intranet.
- *Permission management*. With this function, the knowledge portal administrator can manage the read and write permissions for each user, including the guest users (that is, the Extranet users).
  Read permissions can be defined on an instance basis, which means that the administrator can decide whether a user can visualize an instance or some attributes of the instance. They can be also defined on a concept basis, which means that the administrator can decide whether a user can visualize a concept (and its instances) or some attributes of the concept. By default, any user can visualize all the concepts, instances, and attributes stored in the portal.
  Write permissions are defined on a concept basis, which means that the administrator can decide whether a user can insert, modify or remove an instance of a specific concept. By default, only the administrator can insert concept instances.
- *Ontology management*. With this function, the administrator decides which ontologies are published in the knowledge portal. Any WebODE ontology can be added or removed from the portal.
- *Attribute ordering*. With this function, the administrator can set, for each concept, the order in which the attributes of all its instances will be visualized. Once the administrator has set the order of the attributes of a concept, (s)he can impose this order to the subclasses of the concept.
- *Instance description*. With this function, the administrator can define the set of attributes to be used to describe instances of a concept in the instance list visualization, together with the order in which these attributes will appear. As in the previous case, the description and the order can be imposed to the subclasses of the concept. Figure 6 shows a screenshot of this function while setting the

description of the instances of the concept *Documentation*, where we have selected the *Title* and the *instance description*.

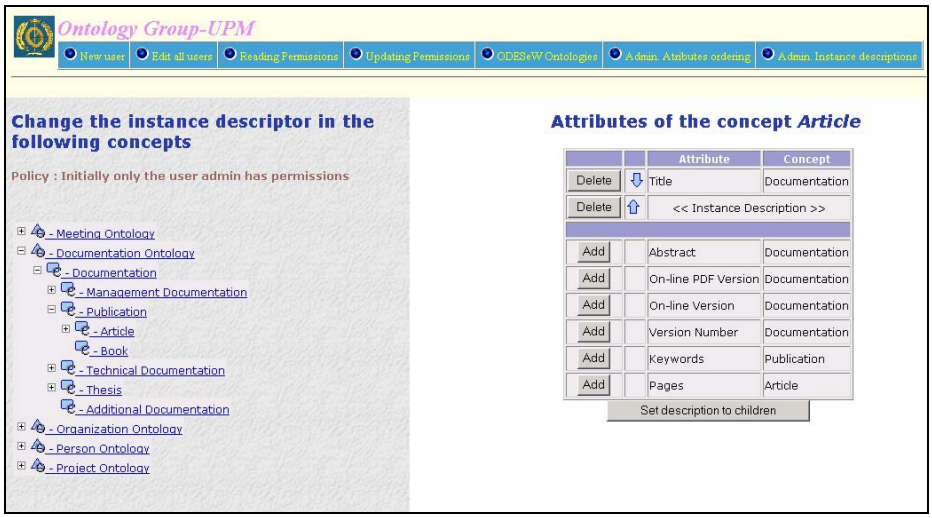

Figure 6. Selection of the attributes that will describe an instance in the instance list visualization.

4. Esperonto web site. A case study of the application of ODESeW

Esperonto [2] (IST-2001-34373) is a European project funded by the European Commission. The aim of this project is to bridge the gap between the current World Wide Web and the Semantic Web by providing a service to "upgrade" existing Web content to Semantic Web content.

The project Web site[2] has been developed as a knowledge portal, powered by ODESeW, with a twofold function: first, to serve as an Intranet for the compilation of all the knowledge generated in the project, and second, to serve as an Extranet for the dissemination of the results of the project.

Five ontologies have been developed in WebODE for this portal: *project*, *documentation*, *person*, *organization*, and *meeting*. They describe respectively R&D projects and their structure, documents that are generated in a project, people and organizations participating in it, and meetings (administrative, technical, etc.) held during a project lifecycle. Figure 7 shows the relationships between all these ontologies (a project has associated meetings, a document belongs to a project, a document summarizes a meeting, people participate in a meeting and have a role in a project, etc.). These ontologies can be reused to describe any R&D project.

These ontologies were defined during the first months of the project life, and were used in the portal. Since then, they have experienced some modifications (adding and

[2] http://www.esperonto.net

removing concepts, adding and removing attributes, etc.). Thanks to the strong interaction of ODESeW with WebODE, these modifications have not caused any consistency problems in the content stored in the portal.

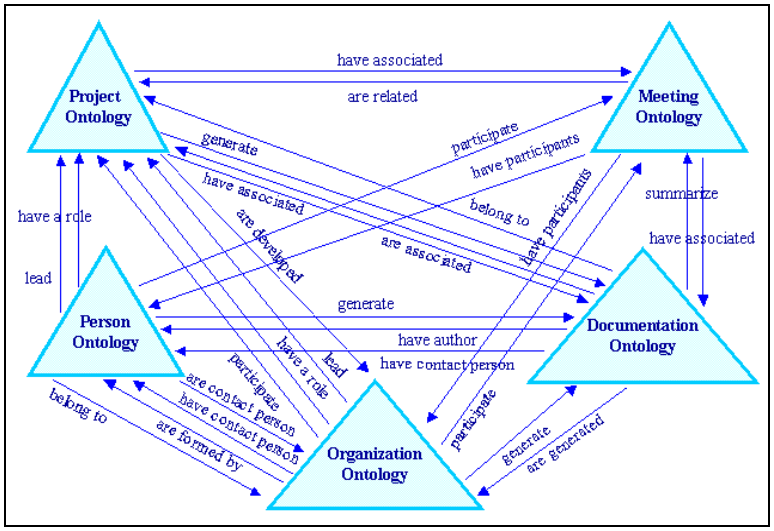

Figure 7. Relations between the Esperonto knowledge portal ontologies.

As of July 2003, the Esperonto knowledge portal contains these 5 ontologies, with around 100 concepts, 200 attributes, 100 relations, 300 instances, which have resulted in more than 100Mbytes of information (this figure includes uploaded files).

We have defined three different visualizations of the Esperonto knowledge portal: the Extranet view for guest users, the Intranet view for the project partners, and a special view in the Intranet for the project officer. For each of these views, we have defined different levels of permissions in the portal. For instance, the guest user cannot insert instances in the portal, cannot access the restricted and private deliverables, can only access the PDF versions of public deliverables, etc.; the Intranet users have full read and write permissions for all the concepts, instances and attributes in the five ontologies; and the project officer view contains automatically generated reports of the project progress.

One of the advantages of using ODESeW on top of the WebODE ontology access service (ODE API) is that it is easy to access to the information in the portal programmatically, and that predefined queries can be easily constructed if needed. In the Esperonto portal home page, which is the only one that is usually modified manually by the portal administrator, we have defined several predefined queries, so as to generate automatically reports of public deliverables (which is the information most commonly looked for by the Extranet users, and even by the Intranet users), summarize the status of deliverables in the project, etc.

Finally, the Extranet of the Esperonto portal has undergone several usability tests by a third party company, which have allowed to improve most of its visualization functions.

5. Related work

In this section we present related work in knowledge portals and languages that could be used to generate portals. We will focus on the similarities and differences with our approach.

A similar knowledge portal is the **OntoWeb portal**[3] [9], which is used as a dissemination tool of the European thematic network OntoWeb. Knowledge in this portal is structured according to one ontology, which contains information about organizations, persons, documentation, events, etc. There are two ways of inserting content in this portal: by means of forms and by syndicating content annotated in external web resources. In contrast with ODESeW, the OntoWeb portal provides a workflow for publishing information, it is supervised by a privileged user (a reviewer), and provides a syndicator system. However, it works only with one ontology, all the users have the same view on the content stored in the portal and there are not such advanced permission management functions.

The **OntoWebber** [12] is a tool to build portal based on ontology and it was used to build the Semantic Web Community Portal as part of the OntoAgents project [4]. This tool take the sources from ontologies in RDF or UML/XMI, or data based on HTML using corresponding data translator for these two last types of sources. In addition of the ontology domain of the portal, OntoWebber has other ontologies that's defines the portal site: maintenance (rules for content maintenance), personalization (personalize content according the needs of users), presentation (look-and-feel of the Web pages), content (rendering Web pages with the ontology information) and navigation (links between Web pages) schemas. All these ontologies with a query engine generate a portal with statically or dynamically information. This tool can generate a portal fitting with the web developer necessity with all site-view ontologies and powered by a rule engine, but all information is centralized in the local server in contrast with ODESeW in which all functionality can be access remotely by web.

The **OntoRoadMap** portal[5] has been also developed in the context of OntoWeb. It includes six ontologies that describe ontology tools, languages, methodologies, applications, events and business scenarios. These ontologies were developed with WebODE and translated to a relational database schema, so that the portal was built on top of them. The main difference with respect to ODESeW is that the portal and the ontologies are not synchronized, in the sense that modifications in the ontologies cannot be seen at run-time in the portal.

KAON portal[6] is a tool that allows building ontology-based portals, based on the SEAL framework (SEmantic portAL) [8]. Once the ontologies and the visualization

[3] http;//www.ontoweb.org
[4] http://www-db.stanford.edu/Ontoagents/
[5] http://babage.dia.fi.upm.es/ontoweb/wp1/OntoRoadMap/index.html
[6] http://kaon.semanticweb.org/

models for them have been built, KAON portal generates the ontology-based portal using HTML pages. In contrast with ODESeW, the KAON portal has a syndicator to acquire information from several sources. However, its main disadvantage is that whenever information is updated from the sources or the ontology is modified, the modifications are not seen in the generated portal at run-time, but it has to be regenerated again.

There are also two languages that deserve special attention, since they could be used for the generation and visualization of knowledge portals, apart from other kinds of portals: RSS and XTM.

RSS[7], also known as RDF Site Summary, Really Simple Syndication or Rich Site Summary, is a RDF-based language for describing news or other Web content that is available for distribution or syndication from a web site. Several tools based on RSS are available, which are mainly aimed at dynamically generating and syndicating news in a web site.

XML Topic Maps[8] (XTM) is an XML-based language that provides a model and grammar to represent the structure of information resources used to define topics and the associations between topics. The topic map paradigm was fully formalized as an ISO International Standard, ISO/IEC 13250:2000. Like with RSS, there are several tools for web visualization of topics and portals based on topics.

6. Conclusions

In this paper we have presented ODESeW, an ontology-based application built on top of the WebODE ontology engineering workbench that creates automatically knowledge portals that can be used as Intranets and Extranets.

ODESeW provides functions for content provision, content visualization, and content search and querying. It also provides an easy-to-use tool suite for the administration of the generated knowledge portals.

Due to its integration in WebODE, we have seen that any modification in the ontologies published in the knowledge portal can be seen at run-time in it, in contrast with other knowledge portal generation tools, and that any set of ontologies implemented in RDF(S), DAML+OIL, and OWL, among others, can be easily included in ODESeW to generate instantly a new knowledge portal. Besides, ODESeW allows establishing read and write permissions for all the content stored in it. Finally, it provides different visualizations for Intranet and Extranet users, so that Extranet users do not have the feeling that they are using a knowledge portal, but a conventional one.

We have also presented how we are using ODESeW as the Intranet and Extranet of the European project Esperonto.

Some of the aspects of our future work in ODESeW will be focused on (a) including better natural language generation for the instance detailed descriptions in the Extranet; (b) adding more editing functions for content provision, such as transferring instances from one concept to another; (c) providing configuration

[7] http://web.resource.org/rss/1.0/

[8] http://www.topicmaps.org/xtm/1.0/

management and evolution support for the content stored in the portal; (d) and improving the current news system by integrating it with ontological information and with RSS technology.

Acknowledgements

This work has been supported by the Esperonto project (IST-2001-34373), by two research grants from UPM ("Becas asociadas a proyectos modalidad B") and by a research grant from MEC (AP-2002-3828).

References

1. Arpírez JC, Corcho O, Fernández-López M, Gómez-Pérez A (2001) WebODE: a scalable ontological engineering workbench. In: Gil Y, Musen M, Shavlik J (eds) First International Conference on Knowledge Capture (KCAP'01). Victoria, Canada. ACM Press (1-58113-380-4), New York, pp 6-13
2. Benjamins, VR., Contreras, J., Corcho, O., Gómez-Pérez. A. (2002). *Six Challenges for the Semantic Web*. In KR2002 Semantic Web Workshop. April 2002.
3. Brickley D, Guha RV (2003) *RDF Vocabulary Description Language 1.0: RDF Schema*. W3C Working Draft. http://www.w3.org/TR/PR-rdf-schema
4. Dean M, Schreiber G (2003). *OWL Web Ontology Language Reference*. W3C Working Draft. http://www.w3.org/TR/owl-ref/
5. Fernández-López M, Gómez-Pérez A, Juristo N (1997) *METHONTOLOGY: From Ontological Art Towards Ontological Engineering*. AAAI Symposium on Ontological Engineering (Stanford, 1997).
6. Karvounarakis G, Christophides V, Plexousakis D, Alexaki S (2000) Querying community web portals. Technical report, Institute of Computer Science, FORTH, Heraklion, Greece. See http://www.ics.forth.gr/proj/isst/RDF/RQL/rql.pdf
7. Lassila O, Swick R (1999) *Resource Description Framework (RDF) Model and Syntax Specification*. W3C Recommendation. http://www.w3.org/TR/REC-rdf-syntax/
8. Maedche, S. Staab, R. Studer, Y. Sure, and R. Volz. (2002) SEAL -- Tying up information integration and web site management by ontologies. IEEE-CS Data Engineering Bulletin, Special Issue on Organizing and Discovering the Semantic Web, March 2002.
9. Spyns P, Oberle D, Volz R, Zheng J, Jarrar M, Sure Y, Studer R, Meersman R (2003). *Ontoweb - a Semantic Web Community Portal*. Fourth International Conference on Practical Aspects of Knowledge Management (PAKM), 2-3 December, 2002, Vienna, Austria, pp. 189-200, Publishing Year: 2002
10. Staab S, Angele J (2000) AI for the Web - Ontology-based Community Web Portals. 17th National Conference on Artificial Intelligence and 12th Innovative Applications of Artificial Intelligence Conference (AAAI 2000/IAAI 2000), Menlo Park/CA, Cambridge/MA, AAAI Press/MIT Press.
11. van Harmelen F, Patel-Schneider PF, Horrocks I (2001). *Annotated DAML+OIL* (March 2001) Markup Language. Technical Report. http://www.daml.org/2001/03/daml+oil-walkthru.html
12. Yuhui Jin, Stefan Decker, Gio Wiederhold. OntoWebber: *Model-Driven Ontology-Based Web Site Management*. The 1st International Semantic Web Working Symposium (SWWS'01), Stanford University, Stanford, CA, July 29-Aug 1, 2001.