A Framework for Ontological Description of Archaeological Scientific Publications

Andrea Bonomi, Glauco Mantegari and Giuseppe Vizzari
Department of Informatics, Systems and Communication, University of Milan–Bicocca
Via Bicocca degli Arcimboldi 8, 20126 Milano, Italy
{andrea.bonomi, glauco.mantegari, giuseppe.vizzari}@disco.unimib.it

Abstract— **This paper describes the results of the first step in the development of a comprehensive project aimed at realizing a portal and a set of advanced services supporting the sharing of knowledge about Prehistory and Protohistory in the Italian context. In particular, one of these services is represented by a digital library, whose entries (i.e. bibliographic descriptions of publications) will be ontologically described. The paper introduces the approach that was adopted to support the acquisition and representation of ontological knowledge. The software modules that were developed to support these phases allow on one hand the management of the assertional component of the ontology, and on the other the association of the related entities to digital library contents. These descriptions will be exploited to support effective strategies for bibliographic information retrieval as well as semantic navigation schemes through the recommendation of contents related to the currently viewed one.**

I. Introduction

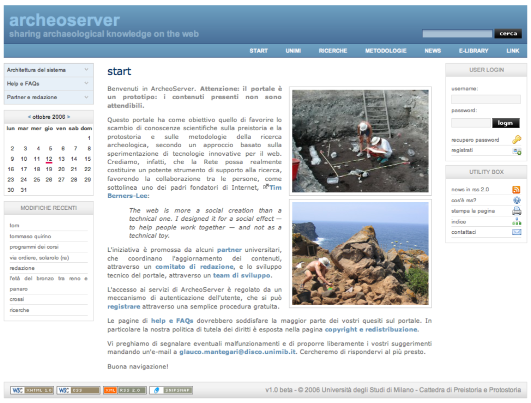
The term e-Science is used to describe science performed through global collaborations between scientists, enabled by Internet technologies, in order to solve scientific problems [1]. Today, no researcher can work isolated but his/her work depends on the available resources in the scientific community. Publications provide one of the main channels for the dissemination of scientific results and it is very important to have access to the right publications when they are needed. Moreover, in most scientific fields, the amount of publications is growing exponentially [2] and finding the right information is correspondingly getting harder. The growth of the amount of existing scientific publications is not a new phenomenon: in the 1960, Maron and Kuhns reported the the fact that documentary data are being generated at an alarming rate, doubling every 12 years [3].

Today there are many on-line digital libraries helping users in finding information about scientific publications. There are mainly two approaches to populate these libraries: manually edit its contents and automatically populate it. The first approach is generally used by libraries, publishers, editors, laboratories, researchers, and so on, whereas the other is often adopted by Internet portals and general-purpose search engines, like Google-Scholar[1] or CiteSeer[2]. These search engines actively retrieve new documents and automatic tags and link metadata information related in a scientific publications structure, without any human intervention. On the other hand, the libraries that use the first approach are maintained with massive human effort, justified only if the offered quality is much higher. For example, in DBLP[3](Digital Bibliography & Library Project) the entries and the related information (authors, conferences, journals, etc.) are manually standardized to guarantee than every entity is always represented by the same string (e.g. every author name is always spelt the same way). This process is necessary because different bibliographic information sources provide information in different formats (e.g. some sources give full authors names, in others names are abbreviated).

Another difference between the two approaches regards the description of the publications contents. The manual approach consists in (manually) associating publications to keywords from a dictionary or a classification system. General classification systems are available (e.g. the ACM Computing Classification System[4] or the Dewey Decimal Classification System[5]), however they are extremely generic and they do not support the description of relations between different publications (e.g. to describe that a publications is part of a collections or to define links between an article and a technical report). The automatic approach consists in extracting keywords from the full-text documents and associating them to the related publication description. This approach requires to have access to the full-text documents in processable form (i.e. if the documents are digitized from hardcopies they must be processed by means of an OCR tool).

This paper describes instead a manual description approach adopted in the specific domain of Archaeology. This work is set in the wider context of a project aimed at realizing a portal and a set of advanced services supporting the sharing of knowledge about Prehistory and Protohistory in the Italian area. For this specific activities partners in Archaeology Departments participate to the project by providing their domain knowledge, but also providing the active participation of (thesis, master, PhD) students that can carry out document description activities. However, in order to effectively describe contents beyond a keyword based approach, and thus in order to support effective forms of information retrieval and semantic

[1]http://scholar.google.com/
[2]http://citeseer.ist.psu.edu/
[3]http://dblp.uni-trier.de/
[4]http://www.acm.org/class/
[5]http://en.wikipedia.org/wiki/Dewey_Decimal_Classification



start

Benvenuti in ArcheoServer. **Attenzione: il portale è un prototipo: i contenuti presenti non sono attendibili.**

Questo portale ha come obiettivo quello di favorire lo scambio di conoscenze scientifiche sulla preistoria e la protostoria e sulle metodologie della ricerca archeologica, secondo un approccio basato sulla sperimentazione di tecnologie innovative per il web. Crediamo, infatti, che la Rete possa realmente costituire un potente strumento di supporto alla ricerca, favorendo la collaborazione tra le persone, come sottolinea uno dei padri fondatori di Internet, **Tim Berners-Lee:**

The web is more a social creation than a technical one. I designed it for a social effect — to help people work together — and not as a technical toy.

L'iniziativa è promossa da alcuni **partner** universitari, che coordinano l'aggiornamento dei contenuti, attraverso un **comitato di redazione**, e lo sviluppo tecnico del portale, attraverso un **team di sviluppo**.

L'accesso ai servizi di ArcheoServer è regolato da un meccanismo di autenticazione dell'utente, che si può **registrare** attraverso una semplice procedura gratuita.

Le pagine di **help e FAQs** dovrebbero soddisfare la maggior parte dei vostri quesiti sul portale. In particolare la nostra politica di tutela dei diritti è esposta nella pagina **copyright e redistribuzione**.

Vi preghiamo di segnalare eventuali malfunzionamenti e di proporre liberamente i vostri suggerimenti mandando un'e-mail a **glauco.mantegari@disco.unimib.it**. Cercheremo di rispondervi al più presto.

Buona navigazione!

v1.0 beta - © 2006 Università degli Studi di Milano - Cattedra di Preistoria e Protostoria

Fig. 1. A screenshot of the ArcheoServer Home Page

navigation, human annotators must have available a domain ontology whose elements can be selected as relevant indicators of the topics treated in the described publication. The paper introduces the ontological description approach, as well as the software modules developed to support the definition of the archaeological domain ontology and the e-library document annotation.

The remainder of this paper is organized as follows. Section II provides a description of the application scenario which is followed by a discussion of related works. In Section IV we discuss the chosen content description approach and in Section V we describe the overall system architecture. We will end with an outlook about possible future extension.

II. Application Scenario

In the course of 2005, the chair of Prehistory and Protohistory of the University of Milan, in collaboration with the Department of Informatics, Systems and Communication of the University of Milan-Bicocca and the Department of Archaeology of the University of Bologna, have started a long-term project for the creation of a set of Web-oriented services aimed at supporting the sharing of knowledge on prehistory and protohistory in Italy.

The main objective of the project has been the creation of a Web portal, named ArcheoServer[6], which will provide a collaborative platform for the exchange of scientific information among the communities of Italian archaeology researchers. A first type of information regards the preliminary results of the research in progress (e.g. that relating to the archaeological excavations conducted during the year), which are rarely communicated to the scientific community before being revised and included in larger studies. Moreover, the portal will provide an easy access to more articulated and analytical contributions on specific topics (e.g. those discussed in a PhD thesis or in a article in a scientific journal), by means of the electronic publishing of traditional papers.

A particularly relevant section of the portal is devoted to a *e-Library* which was devised to supply an effective mechanism for the retrieval of digital resources related to prehistory and protohistory, and in general to the archaeological research methodologies. In fact, even if there are a growing number of initiatives providing for the electronic publishing of scientific papers - as, for example, the digital archives of the Italian Institute for Prehistory and Protohistory[7] or the BibAr[8] project hosted at the University of Siena - their indexing by traditional search engines is often unsatisfactory.

The main requirement of the portal is to give the community itself the possibility of autonomously managing the contents by means of simple editing tools. At this regard, we must keep in mind that, in most cases, archaeologists have just low-level technical competence and the development of a complex editing system may result in the failure of the project.

In our scenario there are two principal classes of editors. The first one is represented by the students of Archaeology of the Universities involved in the project, who are responsible of the content creation; the second one is represented by Archaeology professors or researchers that, beyond creating contents, supervise the work of students.

However, our intent is also to create a platform for the experimentation of computer science research, focusing on those aspects that can lead to a real innovation, such as the semantic description and retrieval of the contents. In fact, scientific publications in archaeology reflect its strong interdisciplinar nature in terms of contents richness and articulation. For this reason it may be interesting to describe, by means of a specific ontology, all the publications that will be archived in the e-Library section in order to provide advanced instruments for a more effective retrieval of the specific information a user is interested to. Moreover the system may even suggest relevant contents which are semantically related to the ones the user is actually viewing on the screen.

Therefore, the e-Library must allow content editors to describe semantically all the publications in a collaborative and simple way, by adopting a simple web-based user interface. In particular this description will be performed manually by the students, while archaeology professors and researchers will supervise the work and will progressively refine/maintain the domain ontology.

Since ontologies are complex to build and understand, the ontology terminological component (roughly speaking, the structure of the ontology) has to be designed by archaeology professors and researchers with the aid of knowledge engineers. In our scenario, since after the initial design of the ontology a structural modification occurs rarely, an ontology editing tool, external to the e-Library Web-based system, can be used for this activity. The e-Library has only to support the maintenance on the domain ontology assertion component (the instances of the concepts defined in the terminological component). Figure 2 shown a scenario that conveys how different users groups should interact with the application main components.

[6]http://www.archeoserver.it/

[7]http://www.iipp.it/

[8]http://www.bibar.unisi.it/

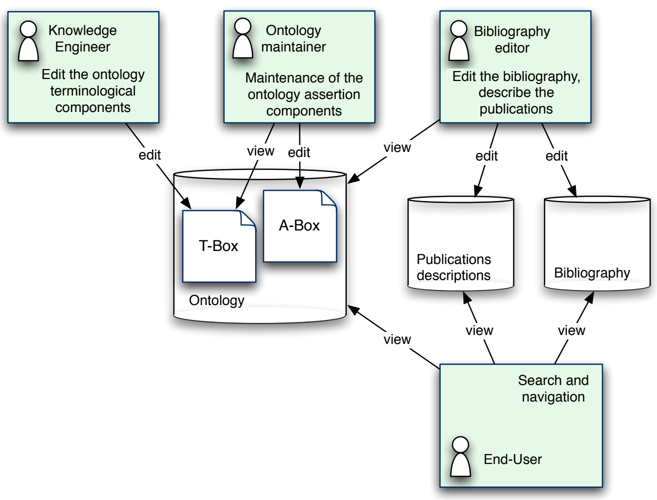

Knowledge Engineer
Edit the ontology terminological components
Ontology maintainer
Maintenance of the ontology assertion components
Bibliography editor
Edit the bibliography, describe the publications
edit
view
edit
view
edit
edit
T-Box
A-Box
Ontology
Publications descriptions
Bibliography
view
view
view
Search and navigation
End-User

Fig. 2. Use case displaying users groups and their actions over the system

A non-functional requirement for the e-Library is the adoption of OWL[9] (Web Ontology Language) as ontology language to describe the publications contents. OWL was adopted because it allows representing and exporting ontological knowledge in an interoperable way.

It must be noted that this paper reports the first results of the project, but we also aim at adopting this approach to the ontological description of other contents of the portal, from images depicting findings and sites, to specific elements of interest in the webGIS (e.g. sites, settlements). The description of these aspects of the project, however, are out of the scope of this paper.

III. Analysis of the Related Systems

Before choosing how to develop the e-Library, different available bibliography information system, semantic annotation frameworks, OWL editors and viewers have been analyzed. A summary of the analysis of such systems will be given in this section.

The e-Library is mainly inspired by DBLP[10](Digital Bibliography & Library Project). DBLP is a Computer Science Bibliography developed by the University of Trier that allows searching a huge collection of bibliographic information (in October 2006, more than 800.000 publications) with a easy-to-use Web interface. The Web interface also allows browsing the bibliography by following links of author, citations, journals and conferences. DBLP collects bibliographics information provided by publishers, editors and so on. A detailed description of DBLP, its architecture, evolution and perspectives can be found in [4]. Since DBLP was started as a prototype Web application in 1993, several years before the birth of the Semantic Web initiative, it does not provide any form of semantic description of the publications.

[9]http://www.w3.org/TR/owl-features/
[10]http://dblp.uni-trier.de/

CiteSeer[11] in another example of a Web-based scientific literature digital library which was developed by the NEC Research Institute. The aim of the project is not only to create a digital library but to provide algorithms, metadata, services, techniques, and software that can be used in other libraries. CiteSeer offers to the users features similar to DBLP but uses a different approach to populate the library: CiteSeer actively retrieves new documents and automatic tags and links metadata information inherent in an academic documents syntactic structure [5]. In our opinion, CiteSeer offers many interesting features, but since it is not an open source product, we cannot use it for our e-Library framework.

Another application for assisting users in managing, searching, and sharing bibliographic information is Bibster[12] [6]. It allows searching bibliographic information on a distributed peer-to-peer network using Semantic Web technologies and provides an easy way to share data with other users. Bibliographic data are represented following the SWRC (Semantic Web Research Community) Ontology [7]. This ontology defines a shared and common domain theory that represents a research community, its researchers, topics, publications, tools, and relations between them. However, Bibster does not match our requirements since the project requires a web-based e-Library application.

Out of the bibliographic domain, there are many ontology-based Web search applications which we have analyzed. OntoWeb [8] is a semantic portal through which knowledge can be gathered, stored and accessed by members of a certain community. Knowledge retrieval and extraction is based on the documents ontological annotation. In the portal, the hierarchical organization of the different concepts of the ontology is graphically represented as a dynamic tree, from which the users can view instances of a class by expanding the tree nodes and selecting the element of interest. OntoWeb graphically displays only the relations of the classes but not the relations between individuals. In our opinion, this kind of visualization is not suitable for our requirements, because the relation between specific e-Library contents (i.e. individuals) are extremely relevant.

Sesame[13] [9] is an open source framework with support for inferencing and querying on RDF and RDF Schema. Despite it is mainly a library for building applications that need to work with RDF, Sesame comes with an interface to allow access to semantic repositories through a Web browser. The interface supports both semantic query and navigation of the ontology via hyperlinks. However this interface is not intended to support end-users with little or no knowledge about ontology languages and thus it offers only a basic support for our requirements. From the developer point of view, the API provided by Sesame are comparable to the Jena API. In an evaluation of different knowledge base presented in [10], Sesame seems to be faster than Jena. However we choose

[11]http://citeseer.ist.psu.edu/
[12]http://bibster.semanticweb.org/
[13]http://www.openrdf.org/

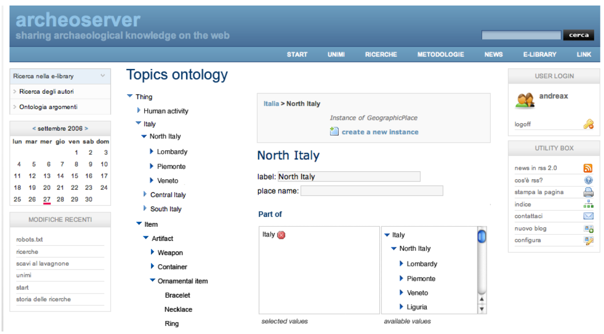


Fig. 3. A screenshot of the A-Box Editor

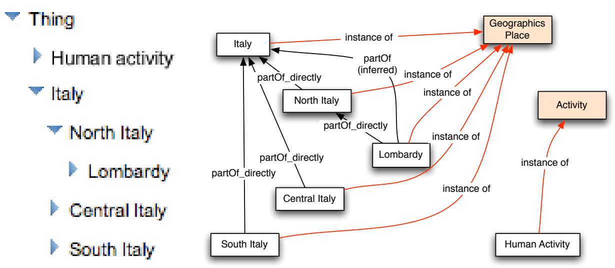


Fig. 4. Navigation tree and graphical representation of the correspondent ontology graph

Jena for developing out framework because Sesame lacks a complete support of OWL.

In order to develop the e-Library user interface, many ontology editors and visualization tools have been investigated. In our opinion, these applications are critical because the diffusion of Semantic Web technology depends on the availability of convenient and flexible tools for editing and browsing ontologies.

The more popular ontology editor is Protégé[14]. It is a free, open source ontology editor and knowledge-base framework. A detailed description of Protégé is out of the aim of this paper and can be found in [11]. In our opinion, Protégé is one of the best OWL editor, but its user interface is too complex for a user with no experience of OWL and lacks some useful functions like the inspection of the elements via hyperlinks and comfortable edit/visualization facilities for the A-Box [12].

An interesting Web-based OWL ontology exploration tool is OntoXpl, which is described in [12]. In particular, an interesting features of OntoXpl is the visualization facility for A-Box, that can be displayed as tree whose nodes are individuals and arc are properties. This kind of visualization is suitable for A-Boxes with many individuals. OntoXpl also supports the inspection of the ontology elements via hyperlinks. OntoXpl has inspired the design of the framework user interface, particularly the navigation tree and the A-Box Editor.

IV. Content Description Approach

Following the previously introduced requirements, three e-Library user groups have been identified: ontology maintainers, content editors and end-users. End-users can have no knowledge about ontologies and related editors, and ontology maintainers are supposed to have a limited background of ontologies. Thus, one of the most important decisions in the design of the e-Library is how to display and edit the ontology terminological component (a set of classes and properties, in the following called T-Box) and assertion component (a set of T-Box-compliant individuals, in the following called A-Box) in a user-friendly way.

As mentioned in Section II, in this framework, there is no specific tool to edit the T-Box. We adopted Protégé to edit the ontology terminological component, save it as an OWL file, and than we import this file in the framework.

From the user's point of view, the developed framework is composed of different modules: A-Box Editor, Publication Description Interface and End-Users Interface. This last module is divided in three submodules: the Semantic Query Interface, the Semantic Navigation Interface and the A-Box Viewer. Not all the modules are fully implemented yet, in particular the Semantic Query and Semantic Navigation Interfaces are still under development. In the following paragraphs, more details about each module will be given.

A. A-Box Editor

The A-Box Editor is only available for ontology maintainers and enables them to edit the ontology A-Box.

As shown in Figure 3, the ontology navigation tree is placed on the left part of the interface and the individuals and properties editor on the right. The aim of the navigation tree is to explore the A-Box and select the individual to edit. The navigation tree is not a hierarchy of classes, but rather of individuals connected with *partOf* or *superType*[15] properties. OWL does not contain specific primitives for *partOf* or *superType* properties but it supports suitable mechanisms to express the features we wanted to specify for these properties.

We defined both these properties as *transitive* (e.g. if *Varese Province* is part of *Lombardy*, and *Lombardy* is part of *North Italy*, then *Varese Province* is part of *North Italy*). For each property, we also defined a sub-property which is directed and non transitive (e.g. we defined the property *partOf_directly* as a sub-property of which *partOf*). These properties link directly an individual with its "father" and will be used to build the navigation tree. For example, if we assert that *Varese Province* is directly part of *Lombardy*, a reasoner infers that *Varese Province* is part of *Lombardy* and *Varese Province* is part of *Italy*. A description of this approach to the representation of the *Part-Whole* relation is described in [13].

We decided that the displayed navigation tree should not exactly reflect the structure of the ontology A-Box but rather it should attempt to provide a clear and usable presentation of the ontology to the users. An example of navigation tree

[14]http://protege.stanford.edu/

[15]Our *superType* property is different from the OWL *subClassOf*: in fact the *subClassOf* is a relations between classes, the *superType* is between individuals.

is shown in Figure 4. The root of the navigation tree is the “fake” element *Thing*; it is not actually part of the ontology and it is only a placeholder. Under the tree root node, there are the top-level individuals (e.g. *Human activity* or *Italy*). These individuals are connected to the underlying individuals with *partOf* or *superType* properties (e.g. *North Italy* is part of *Italy* or *Farming* has super type *Human activity*).

The editor of individuals and properties is placed on the right part of the interface. Using this editor, an ontology maintainer can create new individuals related to an existent one by means of a *partOf* or *superType* property (as shown in Figure 5), remove individuals, edit the label of an individual (the displayed name) and edit the related properties.

The properties of each classes are defined in the T-Box. Two types of properties are distinguished: *object property* is a binary relation between two individuals and *datatype property* is a binary relation between an individual and a literal (a primitive type, like string or number). Properties can also have cardinality and range restriction. For example, the class *TypologyOfArchaeologicalObject* has the property *buildOf*. This property has no cardinality restriction (so it can have zero, one ore more values) but *Material* is specified as range (co-domain). For instance, *Sword* is an instance of *TypologyOfArchaeologicalObject* and has the property *buildOf Metal*, where *Metal* is an instance of *Material*.

There are four properties editor defined in the framework:

- *single datatype* allows editing a single literal value, displayed as a single line input box;
- *multiple datatype* allows editing multiple literal value, adding and removing values;
- *single object* allows defining a relation with a single individual, presenting the user a tree for selecting the value; the individuals displayed in the tree are only those that are valid for the property range;
- *multiple object* allows defining relations with multiple individuals; It is similar to the single object editor but allows adding and removing individuals, rather than selecting only one.

B. *Publication Description Interface*

The Publication Description Interface allows the content editors to associate a ontology-based description to the publications.

The publications descriptions are statements (i.e. subject-predicate-object triples) that associate a topic defined in the ontology to a publication. The statements predicate (also called property) defines the relation between the publication (subject) and the topic (object). Examples of properties are *hasTopicCulture* and *hasTopicHistoricalPeriod*. These properties are defined in the ontology T-Box and every property is a sub-property of the generic topic property *hasTopic* (e.g. *hasTopicCulture* is a sub-property of *hasTopic*). Range restriction is used to specify the valid values for the property (e.g. *hasTopicCulture* has *Culture* as range). The Publication Description Interface considers the range restriction allowing only to select the valid individuals as values of every properties. For example, the property *hasTopicGeographics* accepts only instances of *GeographicsPlace* as object, so, as shown in Figure 6, the interfaces only allows to select instances of this class.

C. *End-User Interface*

End-User Interface is composed of three submodules: the A-Box Viewer, the Semantic Query Interface and the Semantic Navigation Interface. Not all the submodules are yet fully implemented.

The A-Box Viewer is directly derived from the A-Box Editor. Through this module users can view ontology individuals and their properties, browse properties via hyperlinks and access related publications thanks to their description. Browsing the ontology is essential for the user to explore the available information and it also helps non-expert users to refine their search requirements, should they start with no specific requirement in mind [14].

The Semantic Query Interface is in an early stage of development. Currently it only allows searching for papers characterized by a specific topic. The interface, as shown

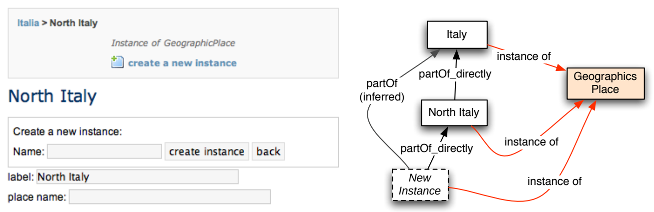


Fig. 5. Dialog box to create a new individual under North Italy and graphical representation of the graph after the creation of the new individual

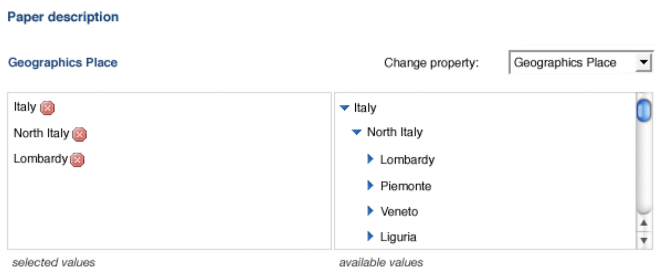


Fig. 6. A screenshot of the Publication Description Interface

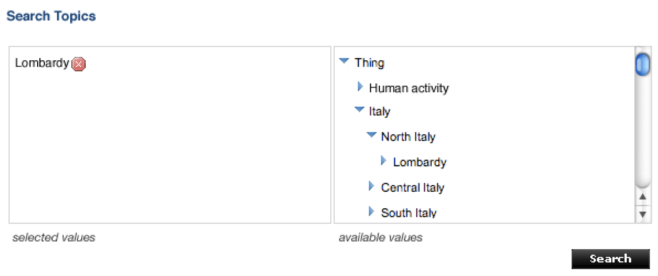


Fig. 7. A screenshot of the Semantic Query Interface

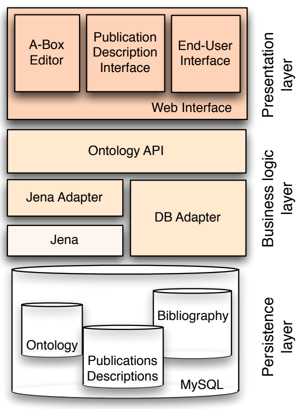


Fig. 8. Overview of the framework three-tier architecture

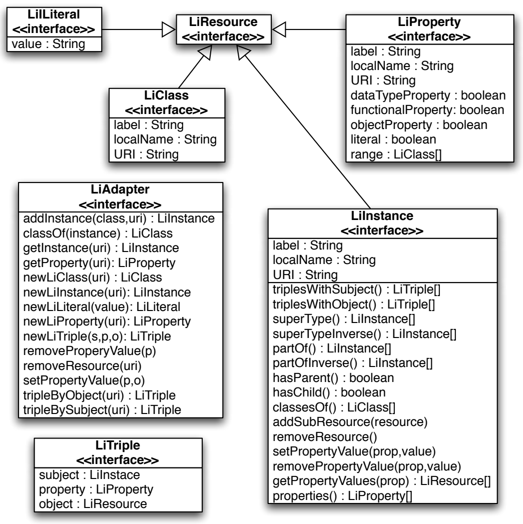

LiLiteral
<<interface>>
value : String
LiResource
<<interface>>
LiProperty
<<interface>>
label : String
localName : String
URI : String
dataTypeProperty : boolean
functionalProperty: boolean
objectProperty : boolean
literal : boolean
range : LiClass[]
LiClass
<<interface>>
label : String
localName : String
URI : String
LiAdapter
<<interface>>
addInstance(class,uri) : LiInstance
classOf(instance) : LiClass
getInstance(uri) : LiInstance
getProperty(uri): LiProperty
newLiClass(uri) : LiClass
newLiInstance(uri): LiInstance
newLiLiteral(value): LiLiteral
newLiProperty(uri): LiProperty
newLiTriple(s,p,o): LiTriple
removePropertyValue(p)
removeResource(uri)
setPropertyValue(p,o)
tripleByObject(uri) : LiTriple
tripleBySubject(uri) : LiTriple
LiInstance
<<interface>>
label : String
localName : String
URI : String
triplesWithSubject() : LiTriple[]
triplesWithObject() : LiTriple[]
superType() : LiInstance[]
superTypeInverse() : LiInstance[]
partOf() : LiInstance[]
partOfInverse() : LiInstance[]
hasParent() : boolean
hasChild() : boolean
classesOf() : LiClass[]
addSubResource(resource)
removeResource()
setPropertyValue(prop,value)
removePropertyValue(prop,value)
getPropertyValues(prop) : LiResource[]
properties() : LiProperty[]
LiTriple
<<interface>>
subject : LiInstace
property : LiProperty
object : LiResource

Fig. 9. UML diagram of the Framework API

in Figure 7, allows selecting the requested topic from the ontology individuals tree. The current implementation retrieves only the publications that satisfies all the specified criteria. A future extension may relax this constraint especially with reference to the number of retrieved publications (adapting the query to the results).

The Semantic Navigation Interface will support users in the e-Library navigation. This system will suggests to the users publications considering multiple strategies for making recommendations (e.g. similar treated topics, recently visited document, user interest, access frequency). This module is not currently implemented in the prototype system and will be object of future work.

V. Framework Architecture

In this section, we introduce a high level overview of the implemented prototype system. The framework was developed according to a three-tier architectural approach, as shown in Figure 8.

The presentation layer is a Web-based user-interfaces. The business logic layer consists of a platform that implements the e-Library main services accessible through a set of API independent from the underlaying storage systems. The aim of the persistence layer is to store the topics ontology, the publications descriptions and bibliographic data.

A. Business Logic Layer

The business logic layer consists of a platform that implements the e-Library main services accessible through a set of API. The main purposes of these API are to support the manipulation and querying of the ontology and the publications descriptions without requiring a detailed understanding of the specific internal storage facility.

The business logic layer interacts with the underlaying layer through a set of adapters: this plug-in interface makes the application independent from the specific implementations. We defined a common API for the adapters: currently implementations of these API are the Jena Adapter and the DB Adapter. The first one is a wrapper for the semantic framework Jena, the other one for the relational database MySQL.

We use Hivemind[16] to develop an open framework that can be easy integrated with new adapters. Hivemind is a framework that supports the configuration of different services, their lifecycle, and their combination. It is inspired by the Service-Oriented Architecture, an approach to the design of software architectures adopting loosely coupled services.

In the framework, the ontology language OWL is used to define a set of concepts and relation between them and to use these definitions to describe the contents of the e-Library publications. OWL defines an information model that can be represented as a directed graph, in which the nodes represent resources and the arcs the properties. The implemented API supports the manipulation and query of these graph in two different ways: *frame-centric* and *statement-centric*.

The *frame-centric* view is similar to the object-oriented paradigm. Every resource is viewed as and object and properties as attribute. This view is used for ontology navigation and resource manipulation. The *statement-centric* is a lower level view in which the graph is represented as a set of triples. Each triple contains three components: subject, predicate and object. This kind of representation is used to obtains query results.

The Figure 9 shows an UML diagram of the framework API. All the information provided to the upper level are modeled using these interfaces. The interfaces *LiClass* and *LiProperty* correspond to OWL Class and Property, *LiResource* represents a generic RDF[17](Resource Description Framework) Resource, *LiInstance* a class instance (an individual), *LiLiteral* a literal and *LiTriple* a statement (an assertion), constituted by a subject, a predicate an a object. *LiAdapter* is the interface of each adapter (i.e. Jena Adapter and DB Adapter).

[16]http://jakarta.apache.org/hivemind/
[17]http://www.w3.org/RDF/

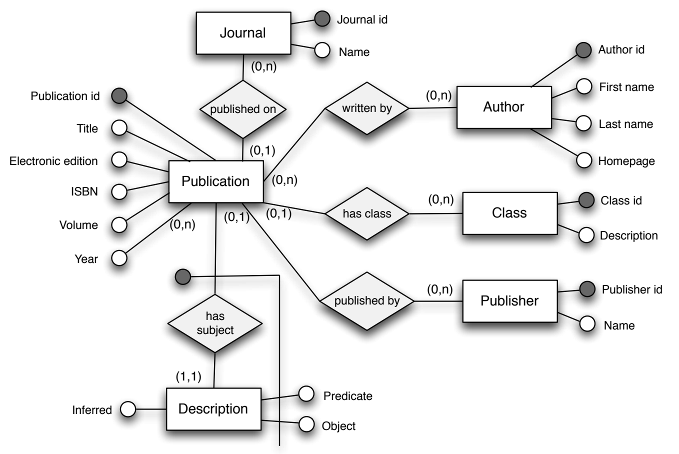


Fig. 10. ER diagram for bibliographics data and publications descriptions

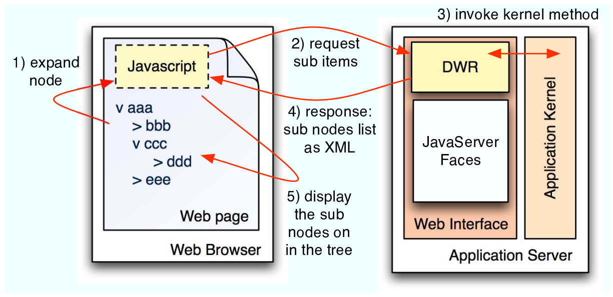


Fig. 11. Interaction among the navigation tree and the server components.

B. Jena

Our choice for OWL ontology storage, manipulation and quering is Jena[18], an open-source Semantic Web Toolkit developed by HP Labs[19]. Its aim is to support the development of applications that use the Semantic Web information models an languages [15]. We have adopted this framework since it matched our requirements and because is widely used within the Semantic Web research community and well documented. The core of the toolkit is the RDF API, which supports the manipulation and querying of RDF graphs (an OWL graph can be viewed as a specialization of a RDF graph, so the Jena API also supports OWL graphs). Jena supports several different storage technologies for ontology persistence. The simplest is to load axioms and individuals directly from an OWL file, but this approach requires the document to be parsed each time the framework starts up and to store after every modification. This can be a source of significant overhead. To avoid this problem, we have used the relational databases persistent storage strategy. This approach also enables faster retrieval and insertion of the ontology elements[20]. To import the OWL ontology created with Protégé into the database, we have used the Jena OWL readers (Jena has readers and writers for different languages that can be used to represent RDF graphs and OWL).

C. Persistence Layer

The topics ontology, the publications descriptions and bibliographics data are stored in on the relational DBMS MySQL[21].

Jena stores ontology in a statements table and other additional tables (e.g. for reification statements); these tables are not intended for direct access by other applications. Publications descriptions and bibliographics data are described in the ontology but they are stored separately for performance issue. Generally speaking, bibliographics data and descriptions can grow very quickly[22] and can have a memory occupation much more relevant than the ontology. If publications were annotated, the number of descriptions could be very high. A semantic framework like Jena, that uses a memory-based reasoner, is not suitable to manage this amount of data (a performance evaluation of several frameworks suitable for large OWL ontologies is presented in [10]).

The bibliographic data and the descriptions are stored in the database, whose schema is shown in Figure 10. The most notable element is the publication description table. This table holds information about publications descriptions as subject-predicate-object triples: the subject is a publication identifier, the predicate defines the type of the topic (e.g. *hasHistoricalPeriodTopic*, *hasCultureTopic*) and the object is the topic of the document. Examples of such triples are: *publication001 hasHistoricalPeriodTopic bronzeAge*, *publication001 hasCultureTopic Etruschi*. According to the defined domain ontology, every publication can have zero, one or more topics, also of the same topic type (e.g. the a publication can be related to both the historical periods Middle Bronze Age and Late Bronze Age).

D. Presentation layer

The main technology used to develop the user interface, described in Section IV, is JSF[23](JavaServer Faces). We choose this technology mainly because JavaServer Faces define a clear separation between application and presentation logic and support the connection of the presentation layer to the application code. JSF defines a set of APIs for representing user interface components, managing their state, handling events, input validation, and defining page navigation.

Another adopted technology is AJAX; it is not a technology in itself, but a term that refers to the combined use of a group of technologies (JavaScript, DHTML (Dynamic HTML)[24], XML and the Remote Scripting) [16]. In particular, we use AJAX for the *dynamic tree component*, that is used as navigation tree, properties editor tree and semantic query topics

[18]http://jena.sourceforge.net/

[19]http://www.hpl.hp.com/

[20]For more information, see: Jena Fastpath Query Processing - http://jena.sourceforge.net/DB/fastpath.html

[21]http://www.mysql.org/

[22]For example, DBLP(Digital Bibliography & Library Project), the Computer Science Bibliography of the University of Trier, indexes more than 800000 publications.

[23]http://java.sun.com/javaee/javaserverfaces/

[24]http://www.w3.org/DOM/faq.html#DHTML–DOM, http://www.w3schools.com/dhtml/

tree. We use AJAX because this technology enables to display new contents in a Web page without completely reloading it. As shown in Figure 11, it is possible to dynamically load the tree elements when required. Having such feature allows it to handle large amounts of data: this is a very important aspect because the tree could be very large and is unnecessary to load all the elements every time.

The AJAX tree is integrated with the rest of the framework by DWR[25](Direct Web Remoting). This technology allows JavaScript code in client Web browser to communicate with the framework running on the server.

[25]http://getahead.ltd.uk/dwr/

VI. Conclusions and Future Developments

In this paper, we presented a prototype of a semantic-based e-Library. This applications allows users searching a collection of publications semantically described. Moreover it gives to the content editors the possibility of autonomously managing the assertional component of the domain ontology, the publications description and the bibliographic data. To describe the publications topic, the e-Library exploits ontology expressed in OWL. A campaign of tests with the students of Archaeology aimed at evaluating the effectiveness of the publications description approach and the usability of the user interface is under way. The tests were focused on the A-Box Editor and the Publication Description Interfaces because these modules are in a more advanced stage of development.

Preliminary results of this tests showed that the proposed ontology visualization is useful for the users as a guide to describe the contents of publications. It helps users with no knowledge about ontologies to understand the relationship between the different topics and between the topics and the publications. Moreover new required features were expressed after the tests. In particular, the users required the possibility to choose the property on which each tree is built on. For example, the users found useful the findings tree build on the “superType” property (e.g. “Sword has super type weapon”, “weapon as super type handwork”), but they can also make use on a tree build on the “hasMaterial” property. Another required feature is the ability to sort the tree items according to a given property. Currently, the items are sorted alphabetically, whereas for some concepts, like the historical periods, this choice is not sensible. For example, the historical periods are better ordered by an explicit “isPrecedent/isSuccessive” property.

The tests also considered the Semantic Query Interface, which is at an early stage of development. Currently it only allows searching for papers characterized by specific topics. The interface allows selecting the topics from the ontology individuals tree and retrieves the publications related with all the selected topics. From the test experience, it might be useful to relax these constraints especially with reference to the number of retrieved publications, adapting the query to the results. For example, if a query selects only a small number of results, the query could be extended to select publication treating also topics related to those explicitly required.

Finally, future works will be focused on the development and test of the Semantic Navigation Interface, which will support users in the e-Library navigation. This system will make recommendations considering multiple strategies: e.g. correlation, recently visited documents, user interests, access frequency. This interface will also capture the cumulative effect of an entire user navigation session in order to generate semantic queries. An description of a work based on this approach can be found in [17].

References

[1] C. A. Goble, “Using the Semantic Web for e-Science: Inspiration, Incubation, Irritation.” in *International Semantic Web Conference*, 2005, pp. 1–3.

[2] M. Ley and P. Reuther, “Maintaining an Online Bibliographical Database: the Problem of Data Quality.” in *EGC*, ser. Revue des Nouvelles Technologies de l’Information, vol. RNTI-E-6. Cépaduès-Éditions, 2006, pp. 5–10.

[3] M. E. Maron and J. L. Kuhns, “On Relevance, Probabilistic Indexing and Information Retrieval.” *J. ACM*, vol. 7, no. 3, pp. 216–244, 1960.

[4] M. Ley, “The DBLP Computer Science Bibliography: Evolution, Research Issues, Perspectives.” in *SPIRE*, Lecture Notes in Computer Science, vol. 2476. Springer, 2002, pp. 1–10.

[5] C. L. Giles, “Citeseer: Past, Present, and Future.” in *AWIC*, Lecture Notes in Computer Science, vol. 3034. Springer, 2004, p. 2.

[6] P. Haase, J. Broekstra, M. Ehrig, M. Menken, P. Mika, M. Olko, M. Plechawski, P. Pyszlak, B. Schnizler, R. Siebes, S. Staab, and C. Tempich, “Bibster - a Semantics-based Bibliographic Peer-to-Peer System.” in *International Semantic Web Conference*, Lecture Notes in Computer Science, vol. 3298. Springer, 2004, pp. 122–136.

[7] Y. Sure, S. Bloehdorn, P. Haase, J. Hartmann, and D. Oberle, “The SWRC Ontology - Semantic Web for Research Communities.” in *EPIA*, Lecture Notes in Computer Science, vol. 3808. Springer, 2005, pp. 218–231.

[8] P. Spyns, D. Oberle, R. Volz, J. Zheng, M. Jarrar, Y. Sure, R. Studer, and R. Meersman, “Ontoweb - a Semantic Web Community Portal.” in *PAKM*, Lecture Notes in Computer Science, vol. 2569. Springer, 2002, pp. 189–200.

[9] J. Broekstra, A. Kampman, and F. van Harmelen, “Sesame: An Architecture for Storing and Querying RDF Data and Schema Information.” in *Spinning the Semantic Web*, MIT Press, 2003, pp. 197–222.

[10] Y. Guo, Z. Pan, and J. Heflin, “An Evaluation of Knowledge Base Systems for Large OWL Datasets.” in *International Semantic Web Conference*, Lecture Notes in Computer Science, vol. 3298. Springer, 2004, pp. 274–288.

[11] H. Knublauch, M. A. Musen, and A. L. Rector, “Editing Description Logic Ontologies with the Protégé OWL Plugin.” in *Description Logics*, CEUR Workshop, vol. 104. 2004.

[12] V. Haarslev, Y. Lu, and N. Shiri, “Ontoxpl - Intelligent Exploration of OWL Ontologies.” in *Web Intelligence*. IEEE CS, 2004, pp. 624–627.

[13] R. Alan, W. Chris, N. Natasha, and W. Evan, “Simple Part-Whole Relations in OWL Ontologies,” Aug 2005. [Online]. Available: http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/

[14] S. Ram and G. Shankaranarayanan, “Modeling and Navigation of Large Information Spaces: a Semantics Based Approach.” in *HICSS*, 1999.

[15] B. McBride, “Jena: A Semantic Web Toolkit.” *IEEE Internet Computing*, vol. 6, no. 6, pp. 55–59, 2002.

[16] G. Jesse James, “Ajax: a New Approach to Web Applications.” [Online]. Available: http://www.adaptivepath.com/publications/essays/archives/000385.php

[17] N. Athanasis, V. Christophides, and D. Kotzinos, “Generating on the fly Queries for the Semantic Web: The ICS-Forth Graphical RQL Interface (GRQL).” in *International Semantic Web Conference*, Lecture Notes in Computer Science, vol. 3298. Springer, 2004, pp. 486–501.

[18] S. A. McIlraith, D. Plexousakis, and F. van Harmelen, Eds., *The Semantic Web - ISWC 2004: Third International Semantic Web Conference. Proceedings*, Lecture Notes in Computer Science, vol. 3298. Springer, 2004.