An Ontology Driven Web Site
and its Application in the Archaeological Context

Andrea Bonomi\textsuperscript{1}, Giuseppe Vizzari\textsuperscript{1}, Bernardo Rondelli\textsuperscript{1}, Sebastian Stride\textsuperscript{2}

\textsuperscript{1} Department of Computer Science, Systems and Communication (DISCo)
University of Milan - Bicocca
\{andrea.bonomi, giuseppe.vizzari, bernardo.rondelli\}@disco.unimib.it

\textsuperscript{2} Department of Prehistory, Ancient History and Archaeology
Barcelona University
sebstride@yahoo.com

Abstract. This paper will describe an approach to the design and implementation of ontology driven dynamic web sites, a kind of architecture that provides the adoption of ontologies rather than more traditional forms of persistent data storage facilities such as relational databases. This approach provides a flexible support to the design and implementation of web portals in which navigation schemes are not entirely predetermined but are instead influenced by actual relationships among the contents of the ontology, that are used to generate web pages as well as hyperlinks. The application of this approach to the realization of a web portal devoted to the sharing of archaeological knowledge about the Silk Road will also be described.

1 Introduction

The introduction of technologies enabling the development of data driven web sites represented an extremely important step in the process that lead the initial version of the Web to its current state of pervasive diffusion and impressive growth rate. Data driven web sites are able to generate web contents and pages according to the information persistently stored in a suitable module, such as a relational Data Base Management System. This represent the last tier of an architecture encompassing a middle tier that is able to interpret suitable templates for web pages that are instantiated according to actual data stored in the data tier, and that are eventually passed to the first tier responsible of the visualization of the page (i.e. the client tier, a common web browser). Traditionally these templates have been realized by integrating modules developed in common programming languages with a web application (e.g. Common Gateway Interface or Java Servlet technologies), or embedding “programming languages–like” control structures and abstractions in traditional web pages through scripting languages (e.g. PHP, ASP, JSP). Some of the most relevant abstractions related to a data driven web site, and determining its organization and function are thus implemented and expressed in terms of a database logical schema and specific programming language control structures embedded in page templates.
Even if this approach surely represents a huge improvement with respect to static web sites, that are essentially repositories of HTML documents and embedded multimedia contents, a set of different research efforts have been carried out in an attempt to supply higher level abstractions to support the design and implementation of web based advanced information systems and applications. Some of these approaches, for instance [1, 2], are based on traditional data conceptual models for site contents and extend the scope of the modelling activity to aspects of relevance in the web context, such as navigation and presentation. In this vein, this paper presents an approach that provides instead the adoption of an ontological rather than data tier, building on experiences and results of research in the Semantic Web [3] area to provide new abstractions and instruments for the structuring and management of information supporting dynamic web pages composition. In particular, the basic idea is to exploit an explicit formalization of concepts related to the web site domain, as well as specific aspects related to web sites in general, to structure data and information required to generate contents, to specify the navigation among them and to generate an effective presentation. The ontological approach provides a uniform and expressive framework for the representation and management of these different aspects.

The following section will elaborate the research context in which this work is set, briefly introducing relevant related works, while Section 3 introduces the architecture and the various functionalities offered by NaVEditOW [4], the framework on which this approach is based. A case study in which the approach has been applied for the development of a portal organizing archaeological information and documents will then be introduced. Conclusions and future developments will end the paper.

2 Ontology driven Web site

In his proposal for a global hypertext [5], Tim Berners-Lee suggested a “gateway program” to generate hypertext view of existing data sources. He has imagined a simply generic gateway with a limited, perhaps read-only, access on a database that would allow it to be displayed as to display it as a hypertext and navigate through the data.

Within a short period, the Internet and World Wide Web have become ubiquitous and today data-driven web sites are ubiquitous (yellow pages, e-commerce sites, digital libraries). But also, forum, blog, wiki, video and photo sharing websites, hotel booking, online auction websites, online community, Web-based email client and Web mapping service are all data-driven.

A data-driven Web site is much easier to maintain than a static Web site: most content changes require no change to the pages. Instead, changes are made to the data - that can also be hosted in an organization information system supporting everyday operations - and those changes are automatically reflected in the web site. Sharing a common data source, the Web application can be easily integrated within the information system. So, for example, in a manufacture company when a new product is added to the enterprise database, it can be automatically displayed on the corporate Web site products catalog.

One of the most common architecture used in developing of dynamic data driven web sites, is shown in Figure 1. In particular, the persistent data storage is generally
Fig. 1. On the left, the architecture of a traditional data-driven Web site and, on the right, the proposed architecture

delegated to a relational database management system. The web server generally hosts dynamic pages including server-side scripts necessary to query the data tier and collect the information required to compose pages related to the contents of the database. These dynamic pages represent a sort of template for the web pages that must be generated according to the stored data, specifying different aspects ranging from the queries that must be submitted to the database to retrieve them, to the kind of links that must be created to support the navigation inside the data.

An evolution of this approach is the adoption of the MVC (Model–View–Controller) [6] architectural pattern. In a web based MVC application, the model is the domain-specific representation of the application information, the view layer enables the model to be rendered into user interface elements (HTML pages) and the Controller processes and responds to the users actions and can invoke data changes. Today there are many MVC framework (written in several languages) for developing of Web based application. Some popular frameworks are Apache Struts, Apache Tapestry, Spring Framework, Ruby on Rails and WebObjects. Apache Struts\(^3\), Apache Tapestry\(^4\)http://tapestry.apache.org/ and Spring Framework\(^4\) are open-source Java Web application frameworks that encourage developers to adopt a MVC architecture facilitating the writing of controllers and templates. Ruby on Rails\(^5\) aims to increase the speed and facilitate the development of data-driven Web sites automatically generating skeleton code for the applications. The fundamental Ruby on Rails principle is “don’t-repeat-your-self” [7]. Rails uses integrated programming packages and preset code designed to be complete and ready to use immediately, without configuration. WebObjects\(^6\) is a development tool and Web application framework. Giving a data model, WebObjects can automatically create a Web application without writing HTML pages and application code.

Generally, the data is stored in a relational database and the model is represented by a database schema. In our opinion, it is difficult to represent all the relations that could be present in a complex domain (such the archaeology domain) with a database schema. In many cases, it is difficult to translate some aspects of a conceptual schema (such the

\(^3\) http://struts.apache.org/
\(^4\) http://www.springframework.org/
\(^5\) http://www.rubyonrails.org/
\(^6\) http://www.apple.com/webojects/
generalization) into the relational model. Some kind of extra-relational constraint are not representable within the database schema and requests database store procedure or external application code. There are also problems to manage the database schema, for example, there is no way to automatically check the schema consistency.

The proposed approach provides the adoption of an ontology [8], rather than a relational database, as data layer (a schema of such architecture is shown in Figure 1).

The main motivations of this architectural modification are related to a more comprehensive exploitation of the explicit relationships among the concepts described in the ontology, and a simpler integration of this kind of architecture with instruments and systems developed in the Semantic Web context.

Other approaches of ontology-driven Web site architecture can be found in the literature: OntoWeaver [9] is an application that uses an ontology to provide a comprehensive support for the design and management of data-driven Web sites. The OntoWeaver approach uses site ontologies to enable a declarative representation of all the aspects of a Web site: in particular, a domain ontology is used as an abstract of the back-end data sources and a user ontology models information about the Web site users. OntoWebber [10] adopts a similar approach in which data from various sources are converted into RDF-annotated format (based on a domain ontology), composed together and used to generate a browsable Web site.

In both cases, ontology provides both an explicit vocabularies for specifying the target web sites and a map for the back-end data sources. Our approach is quite different, since we propose an application for navigating, querying and updating ontological KBs rather then mapping existing databases into an ontology.

3 NavEditOW

In this section we present a system for web based navigation, querying and updating of ontological KBs. The presented software allows the exploration of the concepts and their relational dependencies as well as the instances by means of hyper-links; moreover, it provides a front-end to query the repository with the SPARQL\(^7\) query language.

NavEditOW is an environment for navigating, querying and A-Box\(^8\) editing of OWL\(^9\) (Web Ontology Language) ontologies through a web-based interface.

With respect to ontology navigation, A-Box navigation must be supported because individuals are a fundamental source of knowledge for people accessing an ontology. From this perspective, it is important to support not only the navigation of concept hierarchies defined by isA relations, but also that of other forms of relations of the individuals of each domain. As a first example, locations can be linked through a partOf relation and it should be possible to group locations under the location of which they are all subparts (e.g. browsing all countries of which Europe is composed of starting from Europe); as a second example consider a number of historical periods ordered according

---

\(^7\) SPARQL (SPARQL Protocol and RDF Query Language) is an RDF query language standardized by RDF Data Access Working Group of the World Wide Web Consortium. For more information, see http://www.w3.org/TR/rdf-sparql-query/

\(^8\) The A-Box is the “assertion component” of a knowledge base.

\(^9\) http://www.w3.org/TR/owl-features/
to a relations such as followedBy: it should be possible to exploit this relation to sort
such individuals from the first to the last one.

With respect to editing, although T-box\(^{10}\) maintenance requires a certain knowledge
of ontological formalisms. A-Box editing should be supported taking into account the
following constraints: (i) cardinality and range restrictions defined in the T-Box must
be respected; (ii) ranges of properties and individuals stored in the ontologies can also
be exploited to drive and suggest instance update. Moreover, contextual editing, that is,
the editing of the A-box while browsing the ontology, should be supported.

Although end-users may not be familiar with query languages, the possibility of
performing expressive queries should be supported. On the one hand a language as
similar as possible to well-known query languages for relational databases language
should be preferred. On the other hand, interfaces enabling non expert users querying
the ontology should be developed (e.g. query forms).

In the following paragraphs, we presents more details about each of these tree basic
functionalities and the application architecture.

**Navigation** – With the ontology navigation interface the users can view ontology
individuals and their properties and browse properties via hyperlinks. Browsing the on-
tology is essential for the non–expert user in order to explore the available information
and to refine their search requirements, when they start with no specific requirement in
mind \([11]\). The hierarchical organization of the different concepts and individuals of the
ontology is graphically represented as a dynamic tree. The aim of the navigation tree is
to explore the ontology, view classes and instances, discover the relation between them.
The tree does not represent only a hierarchy of classes connected with isa binary rela-
tions (like the navigation tree of Protége), but represents also tree connections of
individuals for domain dependent classes of properties (e.g. partOf, isLocatedIn, and so
on). From a formal point of view, ontological relations supporting tree-like visualiza-
tion (tree-like properties) are those represented as non symmetric properties and whose
inverse is functional (therefore identifying directed acyclic graphs). These properties
directly link an individual with its “father” and are particularly relevant with respect
to mereological relations (e.g. partOf, composedOf), and to relations defining hierar-
chical spatial and temporal structures (e.g. representing the unfolding of historical pe-
riods). Another kind of relations exploited for the visualization are relations defining
total orders on individuals (e.g. isFollowedBy).

The root of the navigation tree is the OWL class Thing, and the rest of the tree is
organized as follows: under the root node, there are the top-level classes (i.e. direct
subclasses of Thing); each class can be expanded to show its subclass hierarchy and its
individual members; individual-to-individual tree connections are defined according to
a number of selected tree-like properties (e.g. partOf); finally if total order relations are
selected, they are exploited to order individuals within a given level of the tree. In order
to distinguish between classes and individuals, the former are represented in petrol blue
while the latter are shown in shocking pink.

In Figure 3, Geographical Place is a class and Central Asia, Uzbekistan, etc... are
its instances. These individuals, in turn, are connected each other by the partOf,directly
property.

\(^{10}\) The T-Box is the “terminological component” of a knowledge base.
Editing – The application allows the users to create, edit and remove individuals of the ontology, their properties and, in particular, their labels. In fact, to ensure multilanguage support, it is possible to define several labels in different languages for every individual. Users can also create new individuals related to an existent one by means of tree-like properties (e.g. partOf) (as shown in Figure 3): the new individuals are immediately displayed in the navigation tree under their “father”.

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). Cardinality and range restrictions for properties are used to support users while editing. For example, in an archeological ontology, the class `TypologyOfArchaeologicalObject` has the property `builtOf`. 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 `builtOf Metal`, where `Metal` is an instance of `Material`.

The framework includes a datatype and an object editor. The former enables the editing of literal values, displayed as a text input box, whilst the latter enables the definition of property values, which can be selected by the user from a tree. The individuals included in the tree are only those that are valid for the property range.

Querying – The first implemented query interface is the SPARQL query form in which users can write query in the SPARQL language, display results in paginated tabular form and navigate through results via hyperlinks. This interfaces is very flexible because the users can write arbitrary queries but is not suitable for end users. Another kind of query interfaces is based on a predefined set of queries. Every predefined query is composed of a description in natural language, a SPARQL query with eventually
free parameters and a list of parameters. Every parameter is defined by a label, a type and eventually a restriction on the valid values (e.g. a parameter can be filled only with instances of a specific class). For this interface, users can select a query by its description, fill the query parameters and execute it. The results are presented as the results of the other query form. A future extension of this query mechanism may adapt the query with reference to the number of retrieved results.

**Application architecture** –From an architectural point of view, the functionalities (ontology editing, navigation and querying) of the user interfaces are based on the Application API, as shown in Figure 4. The main purpose of this API is to support the manipulation and querying of the ontology through the standard SPARQL query language and through a set of specific adapters, shielding the user from the underlying semantic framework. A plug-in interface, in fact, makes the application independent from the adopted specific semantic framework (as long as SPARQL is supported). A different adapter for every semantic framework is needed because SPARQL is only a query language and does not offer any data manipulation statements (e.g. INSERT, UPDATE, DELETE).

The adapter API supports the manipulation and query of the RDF graphs in two different ways: frame-centric and statement-centric. The former view is similar to the object-oriented paradigm: every resource is viewed as an object and properties as attribute. This view is used for ontology navigation and resource manipulation. 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.

Currently two semantic framework adapters have been implemented: the first one is a wrapper for the Jena Semantic Web Toolkit, the other for the Sesame Framework. The former is an open-source Semantic Web Toolkit\(^{11}\) aimed at supporting the development of applications that use the Semantic Web information models and languages [12]. We have initially adopted this framework since it matched our requirements, it is widely used within the Semantic Web research community and well documented. This first adapter implementation works well with small ontologies, but fails with larger ones since Jena is not suitable to manage an huge amount of data (a performance evaluation of several frameworks suitable for large OWL ontologies is presented in [13]). For this

\(^{11}\) [http://jena.sourceforge.net/](http://jena.sourceforge.net/)
reason, we have chosen to develop a new adapter for the Sesame Framework\(\textsuperscript{12}\). Sesame provides a number of functionalities for handling (querying and manipulating) RDF graphs. It also supports various types of storage facilities and inference mechanisms. The default implementation supports inferencing and querying on RDF Schema but lacks a specific support for OWL. Sesame can be easily integrated with OWLIM\(\textsuperscript{13}\), which is a high-performance semantic repository: in fact, OWLIM can scale to millions of statements according to benchmarks reported in [15]. Both frameworks support integration with external reasoning services.

4 The SilkRoDE Case Study

SilkRoDE (Silk Roads in the Digital Era) is a project that aims to collect, structure and diffuse all knowledge concerning the Cultural Heritage of Central Asia from fields such as archaeology, art history, ethnography, geography, history and sociology. It is an open and evolutive project, which functions as an intelligent network linking all interested institutions, research groups and scholars, that worked or are working in Central Asia.

Regrouping all Human Knowledge on Central Asia may sound more like one of Borges’ dreams than a serious project. However, the idea is not new and various attempts at creating such a system have already been made. Way back in the 1950’s a number of Soviet scholars proposed the creation of a database and a series of maps of all the archaeological and ethnographical material of Central Asia. This idea, clearly formulated by B. A. Litvinsky [16], gave rise to a number of publications such as The Archaeological Map of Kazakhstan [17]. But technical and political reasons meant that the overall project was abandoned. Some 25 years later J.-C. Gardin [18] also emphasized the need for an archaeological atlas of Central Asia and laid a theoretical basis for this work in his many publications on information systems and the development

\(\textsuperscript{12}\) http://www.openrdf.org
\(\textsuperscript{13}\) http://www.ontotext.com/owlim/
of technical means of sharing data through information networks. However, as Gardin himself was well aware, the political situation was not yet ripe (his paper was presented at a colloquium in Dushanbe in 1982, in the midst of the Soviet-Afghan war) and the technical means were only just beginning to appear. SilkRoDE aims to improve and use the technology, now available, in order to achieve the dream of these scholars.

As a first step, SilkRoDE aims to create a Knowledge Management System, used on a daily basis by all specialists of Central Asia, interested members of the general public and those involved in Cultural Management. The creation of this Knowledge Management System (KMS) will be possible thanks to a collaboration between specialists from the Humanities and from Computer Sciences in a very close multidisciplinary context. One of the keys to the success of the SilkRoDE project is the decision of all participating scholars, institutions and research groups to work together as Equal Partners. This does not mean that all resources are simply pooled together but that each resource is clearly associated to the authors and funding agencies that enabled its creation. SilkRoDE thus aims to be a Network rather than a new institution. In this perspective, the NavEditOW system represents the general platform adopted by the Project to organize and manage the various different contents of SilkRoDE Knowledge Management System. The main aim is to share information and knowledge, and all partners should thus be enabled to employ the system to publish, connect personal data and information, and to identify other groups or Institutions working or interested in same themes. This need for a collaborative and collective participation in data and information collection phases lead us to choose a web–based approach to the development of the SilkRoDE KMS.

The introduced approach and the NavEditOW system were adopted to represent and organize basic information about institutions, research groups and scholars actively involved in research in Central Asia, as well as bibliographical data and detailed information about the different categories of Cultural Heritage (e.g. archaeological sites, historical monuments, different types of artifacts, etc.). The platform will be integrated with webGIS: the possible connection between specific entities of the SilkRoDE ontology and georeferred entities stored in a GIS will support the visualization of spatial position and distribution of various entities in dynamically generated maps.

The ontological approach provides the required expressiveness and flexibility to support rich forms of navigation among stored contents. In particular, this approach makes it possible to represent and manage relationships such as “is-a” and “part-of” without flattening the related entities in a single table or splitting them in different tables, as would have been necessary had a traditional relational database been adopted. For instance institutions, research groups and individual scholars are all actors of the SilkRoDE ontology, in other words, they are individuals belonging to classes that are related to the Actor class by an “is-a” relation. It is now possible to define a generic relationship binding an archaeological site to an instance of the Actor class or one of its subclasses, without the need to define different relationships. This flexibility in defining and establishing relationships among individuals will support further types of analysis aimed, for instance, at identifying possible connections among actors that are working on similar or related research issues, or geographic areas, or adopting similar methodologies.
5 Conclusions and Future Developments

The paper has described an ontological approach to the modeling, design and implementation of dynamic web sites. The motivations of this effort, as well as related work and the research context were introduced, and the NavEditOW framework was described, in terms of provided functionalities and architecture. A case study providing the application of the introduced approach and framework was also presented.

Future works are mainly aimed at extending the range of the represented and managed concepts, with particular reference to the various topics that can be used to characterize relevant scientific publications, in an effort similar to the one described in [19]. Moreover, in the medium term, the project will consider the possibility of creating specific wrappers able to export contents complying to the CIDOC Conceptual Reference Model\textsuperscript{14} [20], so as to achieve a high level of interoperability with this relevant standard for cultural heritage information organization.

References


\textsuperscript{14} http://cidoc.ics.forth.gr/