Model-Driven Approach for Semantic Web based-hypermedia Applications

Laura Montells, Susana Montero, Paloma Díaz, Ignacio Aedo
Laboratorio DEI. Dpto. de Informática
Universidad Carlos III de Madrid
Avda. de la Universidad 30. 28911 Leganés, Spain
{lmontell@inf, smontero@inf, pdp@inf, aedo@ia}.uc3m.es

Abstract

The existence of web pages that are described semantically via ontologies and metadata conforming to these ontologies is crucial to bring the Semantic Web to life. In this paper we address the problem of developing semantic web-based hypermedia applications through web-hypermedia design methods. For that, we propose a general approach based on extracting an ontology-based design from a traditional model-driven design, thus allowing hypermedia designers to obtain both domain ontologies and annotated documents in parallel with the application design without requiring extra tasks and expert know-how. This approach is presented through a specific hypermedia design method called Ariadne Development Method (ADM) and its software tool, AriadneTool, that automates a semantic extraction process to provide annotated documents in a format suitable for the semantic web, as RDFS and RDF. Moreover, a semantic web platform has been developed in order to enable the publication of the resulting semantic web-based hypermedia application.

1. Introduction

The Web has turned into a medium for sharing knowledge among people, therefore the major emphasis has been placed on how to present it for human readers. However, the increasing amount of information is leading to an information overload. In order to deal with this continuous Web growth, programs must be able to share and process web resources. This is the aim of the Semantic Web [1]: to attain a web of data that can be both human-readable and machine-processable, thus enabling intelligent access to information.

Representing multimedia content (e.g. voice, video, image, and data) with semantics provided by relevant ontology (or ontologies) has been identified as a key challenge for the semantic web. Annotation systems produce semantically tagged pages using web-based knowledge representation languages, such as RDF\(^1\) (Resource Description Framework) and OWL\(^2\) (Ontology Web Language). These systems\(^3\) require documents in HTML format and specific domain ontologies in order to produce annotation in a manual or (semi-)automatic way. However, the building of these ontologies is a difficult task that requires extensive knowledge (both a knowledge on engineering and a domain expert) and, in most cases, the result could be incomplete or inaccurate. Moreover, the annotation of documents is usually made over existing static pages as an additional task that takes a long time and human effort, and this process may be incomplete or incorrect if the creator is not skilled enough. Therefore, the success of the semantic web depends on the easy creation both of domain ontologies and ontology-based metadata by semantic annotation.

Although this kind of system is still necessary to convert existing web applications to semantic web applications, annotation would be best performed while designing the web application, not after it is implemented. In this way, we can take advantage of implicit and explicit semantic assumptions made during the conceptual modeling of web-based hypermedia applications to directly generate semantic web applications without any additional tasks and expert know-how.

Combining the use of design methods with ontologies according to the Semantic Web provides us with benefits coming from both of these approaches:

- Makes possible to re-use information designs.
- Offers new uses for existing data.
- Allows information sharing between applications.
- Increases the flexibility of systems that will adapt as requirements evolve.
- Reduces the cost and risk of the application design.

\(^1\)http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/
\(^2\)http://www.w3.org/2004/OWL/
\(^3\)http://annotation.semanticweb.org/tools/
• Makes searching content easier using semantic queries on web application with a great amount of information.

Consequently, we propose the coupling of ontologies into the development process of hypermedia and web applications into the conceptual modeling of the existing web design method [12]. This combination allows us to provide web pages with semantic contents (the annotation process) as well as contextual information about the domain knowledge involved (the ontology domain) upon which the application is being designed. Here we present this approach applied to a specific hypermedia design method called Ariadne Development Method (ADM) [3]. We have developed a software framework integrated by a process which transforms hypermedia application modeling into a global, syntactically and semantically interoperable knowledge base in RDF(S) format using a software support tool for ADM, called AriadneTool [13], and a semantic web platform devoted to its visualization. It is important to remark that AriadneTool is not intended to be an ontology editor, but it provides a way to semantically annotate application contents.

The paper is structured as follows. Section 2 describes briefly how hypermedia design methods can integrate semantic content in a natural way. Section 3 describes the Ariadne Development Method phases and AriadneTool. Section 4 presents the framework where we will integrate the semantic annotation functionality and explains how an application domain ontology and a presentation ontology together with metadata conforming these ontologies are extracted from the different products of Ariadne Design Method ready for publication on a particular semantic web platform. Finally, sections from 5 to 7 presents related works, some conclusions including future work and acknowledgements.

2. How to add semantics to Hypermedia Design Methods

In order to allow hypermedia design methods to include metadata about web resources that are specified during the development process, we propose to integrate ontologies in the conceptual modeling phase. A model is an explicit specification of a set of concepts and relationships between them that defines a description language for a specific domain of interest. Just like models, an ontology includes definitions of basic concepts in the domain and the relationships among them but with a different starting point, as stated in [6]. While the former usually has as foundation to get a successful use in an application development, in the latter an epistemological level underlies to express the intended meaning about what is being conceptualized. So ontologies are the tool that may yield a more concise semantic to design models.

In this approach, the idea is to map the concepts and relations of models used in hypermedia design methods to an ontology language. While the former contribute with their graphical support, the latter adds semantic support. All hypermedia design methods such as WebML [2] or RMM [7] are based on formal models to capture the essence of hypermedia applications. Most formal models can be expressed in terms of ontologies languages; in our case ADM is based on the Labyrinth model [4] to explicitly describe the elements which define the structure and behavior of the application and it can be expressed by means of an ontology web language such as DAML+OIL [11] or OWL as described in [12].

From this coupling, hypermedia design methods and the semantic web can mutually benefit. On the one hand, methods can integrate web standards for expressing metadata about web resources and include formal semantics for checking completeness, consistency and correctness of the design with the respect to the method semantics, thus improving the user’s understanding of its use, as in [13]. On the other hand, the semantic web can take advantage of the experience gained from years of research in the hypermedia engineering field through its design methods devoted to obtain well-organized application in aspects of information, navigation, presentation, interaction, personalization and even access control.

In the next section, we present a specific hypermedia design method, the Ariadne Development Method and its implementation, in order to introduce how the underlying semantics of a hypermedia application can be extracted during its design process to produce metadata about information and its presentation applying the approach presented here.

3. The Ariadne Development Method and the AriadneTool toolkit

In a nutshell, the Ariadne Development Method establishes a systematic process composed of three phases as illustrated in Figure 1. The Conceptual Design is focused on identifying abstract types of components, relationships and functions; the Detailed Design is concerned with specifying the system features, processes and behaviors in a detailed
way in which the application might be generated; and, fi-

nally, the Evaluation is concerned with the use of proto-
types and specifications to assess the system usability. Fur-
thermore, each one of them proposes a number of design
products to specify and produce hypermedia and web appli-
cations. The arrows shown in Figure 1 mean that the method
does not impose any kind of sequence among phases, let-
ting developers decide the best way to face their work ac-
cording to their needs. Moreover, the method provides a
number of Validation and Integrity Rules, both at the in-
tra and inter phase level to check completeness, consistency
and integrity among the various design products. Ariadne-
Tool [13] is an environment designed to develop hyperme-
dia applications based on ADM supporting fast-prototyping
in HTML, XML, SMIL and RDF, as well as automatic gen-
eration of documentation about the design process.

We will next explain how AriadneTool extracts knowl-
edge about the application domain and expresses it in
RDF/S format, suitable for the semantic web.

4. Automatic extraction of semantic informa-
tion from the design process

Before explaining the extraction process, we will de-
scribe the architecture of our approach which is depicted
in Figure 2. It is made up three component modules. The
Semantic Generator recollects semantics and presentation
characteristics associated to a hypermedia application de-
designed with AriadneTool. It is implemented as a Java mod-
ule within the tool and uses Jena\(^4\) for creating, modifying
and inferring knowledge about the modeling, and expresses
it in RDFS and RDF format. The Semantic Repository
Manager and the Viewer Module are external applications
that are needed to store and manage semantic information
in order to be presented later to the user. The Semantic
Repository Manager uses the Sesame\(^5\) repository for man-
aging the semantics from the application. It uses RDQL\(^6\)
as query language and MySQL to store the metadata gen-
erated by the Semantic Generator. The Viewer Module is
implemented with JSP\(^7\).

As shown in Figure 2 the semantics extracted are stored
on the semantic repository according to two different points
of view: the data view and the presentation view. The
following subsections briefly describe how AriadneTool
extracts semantics from the different products of ADM
through a simple example about a research group website,
which provides information about its members, research ar-
areas and publications. Moreover, some pieces from the an-
notation produced in RDF and RDFS format are included.

4.1 The data view

AriadneTool extracts the ontology and its instances about
the data of the research group website example from the
Conceptual Design and the Detailed Design, respectively.

\(^4\)http://jena.sourceforge.net
\(^5\)http://openrdf.org
\(^6\)http://www.w3.org/Submission/2004/SUBM-RDQL-20040109/
\(^7\)java.sun.com/products/jsp
This process generates an RDFS file containing the application domain ontology (domain.rdfs) and a file containing the ontology domain instance (dataDomain.rdf).

The Application Domain Ontology is extracted from the following products of the Conceptual Design:

- The Structural Diagram allows us to express concepts and relationships that appear in the application domain by means of composite nodes which are connected to their simple or composite components by means of two abstraction mechanisms: aggregation, which refers to a set of nodes as a whole and generalization, which represents an inclusion relation involving inheritance mechanisms.

Figure 3. Structural Diagram from Ariadne-Tool

On the proposed example we want to represent a research group composed of members, research areas and publications. To represent it on AriadneTool, the designer draws the general structure composed of a node Subject that is an aggregation of the node ResearchGroup and the nodes Member, Research and Publication that are a generalization of the node subject. This representation is shown on the right side of Figure 3. On the left side, Part of the Application Domain Ontology that is extracted automatically from the design is shown. On this ontology generalization is represented with the property: subClassOf and aggregation with a new property with a range which is a sequence.

- The Navigation Diagram specifies the navigation paths and tools that the website is going to offer to the users. Navigation paths are settled among nodes using tagged links which can be uni or bi-directional. In the Member node example we browse both the Publications and the ResearchArea node. Since Subject is a generalization composite, all its components (such as Member, Publication, and so on) will inherit the link information.

Figure 4. Navigation Diagram from Ariadne-Tool

Right side of Figure 4 shows a screenshot of the drawing of the Navigation Diagram captured from AriadneTool. The left side contains part of the Application Domain Ontology that is extracted automatically from the design. On this ontology uni-directional links are represented with a property whose domain is the node source and the range is the target node. Bi-directional links are represented through two properties, alternating source and range.

- The Internal Diagram stores information about the spatial as well as temporal dimension of each of the information elements identified in the structural and navigation diagrams.

Figure 5. Spatial Diagram from AriadneTool

The left side of figure 5 shows the Member node visualization area with its contents represented with white boxed, located and aligned. From an ontological point of view, contents are like ontology properties, they are defined in an independent way and are then tied to different nodes, and thus, they can have different domains. For example, the properties of a group Member
include its photo, name biography and address. On the ontology fragment presented on the right side of Figure 5 each node content is presented as a property whose domain is the node where it is included and the range is a reference to the information that will be included in the node.

The ontology domain instance will be extracted in the next stage, the Detailed Design where the entities specified in the Conceptual Design are transformed into more concrete system elements. It is extracted from the following products:

- The Diagram of Nodes Instances where the nodes defined in the structural diagram are created by means of a number of Node Instances. Thus, the Member node is replicated as many times as needed to represent all group members.

- The Detailed Internal Diagrams where all nodes and contents are fully specified and annotated with their values.

Figure 6 is a part of the RDF/XML encoding for the research group where the instances of Member node appear with its values attached to the ResearchGroup resource. From these metadata described in RDF and the structure of the application domain ontology described in RDFS and generated as explained on previous section, we can develop a semantic web application.

4.2 The presentation view

The application domain ontology and its instances enable us to define concepts, relations and system data information to be processed by software tools. However we need to store information on presentation in order to show the semantic content on a conventional browser. This information is extracted from two detail levels:

- Hypermedia information relative to the model annotate hypermedia content according to the Labyrinth model. On Figure 7(a) we can see a fragment from the Labyrinth model ontology and the data extracted from the research group example. This is a high level description extracted from the different products of the Conceptual Design such as the User Diagram and the Access Table (Figure 8) in which the designer can define presentation rules. Using this information we can decide which user accesses each node (content). Taking as example the knowledge presented in Figure 7(a) we can think about the following question: what user or users has privilege to reach the photo content of the Member node? or what content belongs to each node. This information is extracted from the semantic repository containing ontologies and data, using RDQL as query language as mentioned on section 4.

- Presentation details is a low level description of the layout, size and content type. First we generate an ontology about the presentation and then we extract the data from the Internal Diagram of the Conceptual Design (Figure 7(b)).

5. Related works

Currently, other model-driven approaches such SHDM [10], WSDM [14], OOWS [5] or UWE [9], follow the same strategy as ADM with some differences. As mentioned on previous sections, these approaches use Semantic Web-based languages (e.g. OWL, RDFS, RDF) to specify those relevant conceptual constructs that characterize the meaning of the corresponding Web Application. This specification makes possible the connection of the application to any external potential agent. This way, models can be represented by Semantic Web languages. ADM is different from other methods in the sense of implementation of security mechanisms defining users access policies. It also integrates information relative to the navigation into the presentation high level ontology. On this ontology, ADM stores the nava-
6. Conclusions

This paper has argued how hypermedia design methods can provide semantic contents as well as contextual information about the application domain which are modeling in order to face up the Semantic Web. Mapping models to an ontology language provides us some benefits such as the decrease of the cost and risk of the application design and information sharing between applications. We incorporate semantic content generation using the Ariadne Method. We apply this approach on AriadneTool.

Web designers will now provide the annotation during the Conceptual Design. Compared to currently existing annotation methods, this approach extracts the semantic content implicitly so the designer does not realize the process; no additional expert knowledge is necessary for the data annotation and the ontology and data generated. Finally, this process enables us to improve the consistency during the application design process and to speed it up by making use of the metadata already provided. Also we can browse semantic content on a conventional browser using the visual...
alization module that complements AriadneTool. Finally, this approach establishes a technological framework where the application data and functionality can be presented and shared between different web applications.

To conclude, we are extending the architecture presented here for sharing and reusing semantic content generated in the design process of other applications. In addition we are extending the functionality of AriadneTool for importing ontologies already defined as does HERA [16] and OntoWebber [8]. This will allow to validate designs or begin the design from domains previously defined adopting those concepts that are of utility.

7. Acknowledgments

This work is part of the ARCE++ project (TSI2004-03394) funded by the Spanish Ministry of Science and Education (MEC) and a cooperation agreement between "Universidad Carlos III de Madrid" and "Dirección General de Protección Civil y Emergencias" (Ministry of the Interior).

References


