The ODESeW 2.0 Semantic Web application framework

Oscar Corcho
University of Manchester
School of Computer Science
Oxford Road, Manchester, United Kingdom
+44(0)1612756821
Oscar.Corcho@manchester.ac.uk

Angel López-Cima(*), Asunción Gómez-Pérez
Universidad Politécnica de Madrid
Facultad de Informática. Campus de Montegancedo, s/n.
28660 Boadilla del Monte, Madrid, Spain
+34913367467
{alopez, asun}@fi.upm.es

ABSTRACT
We describe the architecture of the ODESeW 2.0 Semantic Web application development platform, which has been used to generate the internal and external Web sites of several R&D projects.

Categories and Subject Descriptors
H.3.3 [Information Search and Retrieval]: Information Search and Retrieval; D.2.12 [SOFTWARE ENGINEERING]: Interoperability - Data mapping

General Terms: Design

Keywords: Semantic Web, framework, web application.

1. INTRODUCTION
The term application framework is normally used to refer to a set of libraries or classes that are used to implement the standard structure of a type of applications. By bundling a large amount of reusable code into a framework, much time is saved for the developer. Application frameworks are also defined as software components that model and solve a specific type of problem, providing a set of extensible and configurable components and an engine to coordinate and execute them. These components will be extended in a specific problem by developers.

The first definition focuses on the reusability of libraries in groups of similar applications. The second one focuses on the fact that an application framework should provide an integrated support to a set of functions that are common in a set of similar applications. In any case, application frameworks allow reducing the amount of effort needed to develop and maintain software, and are part of the philosophy of rapid application development (RAD).

There are many frameworks available for the development of standard Web applications. Among them we can cite: Turbine, Struts, JSF, Millstone, Wicket, etc. They all provide reusable configurable components commonly used in such applications.

In the context Semantic Web application engineering there are fewer frameworks available, due to the fact that this area is less mature than that of Web engineering. In many cases we cannot talk about frameworks, but about specific applications developed completely from scratch or by reusing some existing components, but without the notion of comprehensive application development frameworks. Some of these emergent frameworks are: the KAON portal, OntoWebber, OntoWeaver, Rhizomik, Duontology, etc.

Most of the applications developed in this area are *knowledge portals* or *semantic portals*, defined as web applications that "*provide the means to select, classify and access, in a semantically meaningful and ubiquitous way, various information resources (e.g., sites, documents, data) for diverse target audiences (corporate, inter-enterprise, e-marketplace, etc.).*"

Though both Web and Semantic Web application development frameworks provide interesting features for the rapid application development, they also share the fact that they are not specialized for the development of domain-specific applications. That is, they only contain generic components that can be included in Web and Semantic Web applications and these components have to be extended by developers when they want to create a specific application in a domain.

From this comment it seems interesting to have also reusable extensions or configurations of such application development frameworks for those types of applications that a set of developers normally have to create. In this paper we are interested in showing how we have configured and extended a Semantic Web application development framework for the creation of the Intranets and Extranets of several European R&D projects (Esperonto [3], Knowledge Web [4] and OntoGrid [5]). The application development framework that we have used is ODESeW, whose earlier version was already described in [2].

2. ODESeW Architecture Outline
The architecture of ODESeW 2.0 is based on the design pattern Model-View-Controller (MVC), which is currently widely used for developing Web applications. This pattern divides functionality among three types of objects (the view, the model and the controller), which are involved in maintaining and presenting data to minimize the degree of coupling between the objects. This paradigm is very useful for developing applications where the same information has several visualisations.

2.1 Data Model
The ODESeW Data Model contains the information that the knowledge portals show and that they use for their management functions. It is divided in two submodels, as shown in figure 1: the Domain Model and the User Model. Both of them are based on ontologies, which are accessed using the WebODE ontology engineering workbench [1] as an ontology server.

The Domain Model is composed of a set of ontologies that describe the application domain. In the case of R&D project applications these ontologies are about projects, organisations, documents, and meetings.

The User Model is an ontology used to specify groups and roles, and to associate read and write permissions to different parts of the Domain Model.

WWW 2006, May 23-26, 2006, Edinburgh, Scotland.
ACM 1-59593-323-9/06/0005.

All these submodels are coordinated by the Data Model Manager, which receives state change requests from the controller and is used to feed the queries made by the views. All the state change requests are filtered by the Permission Layer, which takes into account the user permissions and profile.

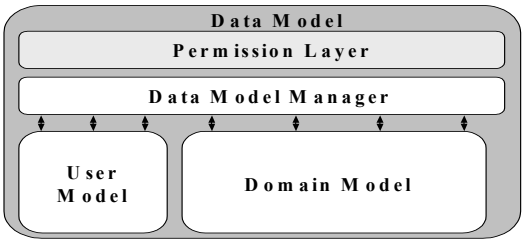

Figure 1. The ODESeW Data Model and its components.

2.2 Views

Views are mainly used to render the content available in the data model. ODESeW provides a set of reusable views and mechanisms for Web developers to ease the communication with the Data Model, so as to retrieve information from the ontologies stored in it. Two groups of views are identified:

- Views for human agents. They are focused on the generation of HTML documents. Some of the reusable views provided are for rendering ontology concepts, instances and relations, as well as their attributes. They use state-of-the-art Web application design technology, such as JSPs, Tag Extension, Expression Language and JavaBeans.
- Views for software agents. They are focused on the generation of documents in Semantic Web languages like RDF, RDF Schema and OWL.

2.3 Controller

The ODESeW Controller receives the user request, which contains the actions to be performed, and completes or checks the request with the information model in the Data Model (including both the domain model and the user model). Then it reads and executes the navigation and composition model, described below, and returns the next view that should be rendered for the user.

The navigation model represents the navigation of a user through the application. This model is explicitly separated from the design of views so that changes in the navigation do not affect the implementation of views. Besides, it allows representing declaratively the navigation of a user, enabling in this way an easy study of the behaviours of the user of an application.

The navigation model is a directed named graph in which nodes represent views (with preconditions) and edges represent navigation actions from one view to another. The model also allows describing specialisation/generalisation relations between two views. A view is a specialisation of another if it visualises the same content as the parent view but providing more specific visualisation items. For instance, a generic view may be used to render any instance from the domain ontologies, while more specialised views could be used to render specific instances, such as instances of a person, organisation, project, etc.

The composition model is similar to the navigation model, and allows including a set of views inside another one. It is normally used when complex sets of information have to be presented to the user at once. A common example of the use of the composition model is the visualisation of an instance. Here the developer specifies that he/she wants to render the value of a set of attributes. The composition model specifies how to render any type of attribute values using generic views and some specialised views for values like e-mail addresses, URLs, image files, etc. All these attribute value views are included in the view used to render instances.

2.4 Extensions to the MVC pattern

ODESeW provides two extensions to the MVC design pattern, both of which are included in the complete ODESeW 2.0 architecture that is depicted in figure 2.

The **External Information Gateway**, used to feed the data model with information available in external information sources, regardless of the communication protocols (HTTP, FTP, CORBA, Web services, etc.) and formats (relational databases, texts in natural language, XML documents, RDF files, etc.) needed to access such information. It supports two information provision models, cached and runtime, which are used depending on the characteristics of the information sources (availability, cost model, processability of information, dynamicity of information, etc.).

The **Notification Service**, used to send asynchronous messages about changes in the data model, following the subscription/notification pattern.

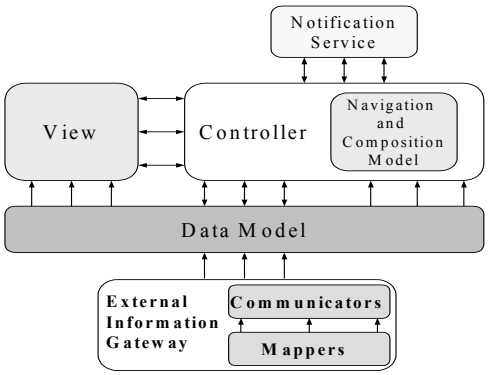

Figure 2. ODESeW extended-MVC design pattern.

3. ACKNOWLEDGMENTS

This work has been supported by the EU IST Network of Excellence Knowledge Web and by the Spanish project TIN-2004-02660. We also thank project partners from this and other projects (Esperonto and OntoGrid) for their useful comments to improve the usability of the sites generated.

4. REFERENCES

[1] Arpírez JC, Corcho O, Fernández-López M, Gómez-Pérez A. WebODE in a nutshell. AI Magazine 24(3):37-48. Fall 2003

[2] Corcho O, Gómez-Pérez A, López-Cima A, López-García V, Suárez-Figueroa MC. ODESeW. Automatic Generation of Knowledge Portals for Intranets and Extranets. LNCS 2870. Springer-Verlag. pp: 802-817. October 2003.

[3] Esperonto. http://www.esperonto.net/

[4] Knowledge Web. http://knowledgeweb.semanticweb.org/

[5] OntoGrid. http://www.ontogrid.net