ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/230651559

A Knowledgebase Editor Using Semantic Web Technologies

Conference Paper · February 2010

DOI: 10.13140/RG.2.2.28622.05443

CITATION
1

READS
78

4 authors:

Goran Bakraceski

3 PUBLICATIONS **1** CITATION

SEE PROFILE

Riste Stojanov

Ss. Cyril and Methodius University in Skopje

33 PUBLICATIONS **97** CITATIONS

SEE PROFILE

Milos Jovanovik

Ss. Cyril and Methodius University in Skopje

58 PUBLICATIONS **184** CITATIONS

SEE PROFILE

Dimitar Trajanov

Ss. Cyril and Methodius University in Skopje

151 PUBLICATIONS **537** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

LDA: Linked Data Authorization View project

Flow2OD: Generation of Universal Simulation Data Based on Real Traffic Data Flow View project

All content following this page was uploaded by Milos Jovanovik on 22 May 2014.

The user has requested enhancement of the downloaded file.

A KNOWLEDGEBASE EDITOR USING SEMANTIC WEB TECHNOLOGIES

Goran Bakraceski, Riste Stojanov, Milos Jovanovik, Dimitar Trajanov

Faculty of Electrical Engineering and Information Technologies
Rugjer Boskovik bb, 1000 Skopje, Macedonia
goran.bakraceski@gmail.com, ristes@feit.ukim.edu.mk,
milos@feit.ukim.edu.mk, dimitar.trajanov@feit.ukim.edu.mk

Abstract

We agree that a semi - structured data model offers the right balance of rich structure and flexible schema allowing users to store and manage data as they prefer, making the access as easy as possible. For that reason we build this system, which exposes the richness and flexibility of the data model, offering natural access to the data and hides the specifics of schema, tuples and database queries. This system uses semantic web technologies for data representation, providing form and text based data entry and retrieval. Data is stored in RDF (Resource Definition Framework) format [1], while the forms are generated from the OWL (Ontology Web Language) data definition [2]. The OWL based guiding is provided for the text entry via suggestions. Users with poor knowledge in OWL and RDF can use this system for defining the format of their data, by entering and altering OWL classes, and manipulate with their instances.

Keywords: silverlight application, semantic web, knowledgebase editor

Introduction

Nowadays one of the most interesting topics in modern web technology is the semantic web and the opportunities that it has. The computers are handicapped when they need to understand information. The information is all around us, but for now, only people can understand it. the semantic way is introducing a new way of knowledge representation, not only for the user,bat to the machines too. The main idea is to create schemas, that will hold the definition of the data used in the application. But, this definition is not as abstract as the class definition in the programming languages. The semantic web formats are designed to be close to the natural language, so that they can be understandable for regular people, and not only by the experts. Thus, the people can easily use and manipulate the data, and define its structure using structure that is very close to their natural language.

Using this new technology, development of new GUI (Graphical User Interfaces) can be semantically driven and users can use and manage data in their preferable way.

However, we have to mention that this technology is quite new and users doesn't have much knowledge for the data representation, even it is similar to our way of communication whit each other. One of the main objectives of this application, among others, is exactly that, to provide easy and simple interface for creating new ontology and changing existing one. For providing the user flexibility in what they store and retrieve, this application communicates with knowledge management system (KMS). Objects are identified by *uniform resource identifier (URI),* which guaranty the uniqueness of the resource. Using this URI, user can reference any resource of its interest, while he builds his own knowledgebase. In that way, users are exempt from learning methods for creating ontologies and schemas, simply because the application provides suitable way for it. On the other hand, with this application we can link the knowledge, offering one resource to be significant for the other.

At the beginning of this paper, we will outline few applications with similar behaviour and goal as ours. They use the advantages and benefits of the semantic web for storing data and creating ontology. Then, we will summarize the primary goals and the motivation for building this application, and present the the web based guided text editor and the form based editor. The paper is closed with a section on related work and some conclusions.

Related Work

The developed web based guided text editor aims to improve user ability for creating and editing ontology, together with their instances, using guided natural language form, with typing natural language sentences. It offers using of auto complete feature that is guiding the user in the data entry, and user can store and retrieve his data with no knowledge in semantic web technology and its formats of knowledge representation. Similar applications, that are also using the advantages of the semantic web, are described by Error: Reference source not foundError: Reference source not found. In GINOError: Reference source not found the author describes the application for guided ontology creation and manipulation using text based interface, which is a Java application for creating new ontology and editing existing one. Here, user can see and select any class or object from the ontology using the application and additionally can edit selected one. The Haystack Error: Reference source not found application is much more complex. It uses semantic data for storing, searching and retrieving data for enabling flexible user interface and resource linking in a way that user wants. This application allows users to customize their interface. Because the User Interface is customizable, this makes users to be more efficient. Resources from different areas can be linked together, making this application replacement for several other applications. User can link its favourite data in this way:

©2010 Institute of Informatics.

- Associate e-mail about a certain interview with the interview article you are writing;
- Link musicians to concerts they played, songs they performed, and photographs they are in;
- "Caption" photographs of musicians with the song being performed in the photo;
- Place songs or albums in a calendar according to release date;

Therefore, the power of using this new way of storing data and its manipulation, was main motivation for building this knowledgebase editor. It functionality can be divided in two sections. In the first section user can create or edit an ontology using forms. The forms are well defined so, users with no experience in semantic web can add new classes and objects, or edit the existing one.

Knowledgebase Editor Usecase

The user interface is designed using Silverlight 2.0 [6], while the data manipulation is done using the Jena ontology framework [8]. Because Jena is developed in Java, the communication between Jena and the user interface is achieved by using web services. The two programs communicate with each other with *common objects*. These common objects are all that we need to get information about ontology classes, properties, instances etc. Currently they are wrapping the Jena defined objects for the corresponding instance. We use four types of objects including CommonClass, CommonProperty, CommonInstance and CommonStatement. These are sufficient to make the necessary operations. The architecture and data model will be described in details in the next section.

At the application startup, all classes and properties from the ontology are loaded into the Jena model, so the user can manipulate with them. The interface for data manipulation is shown in Figure 1.

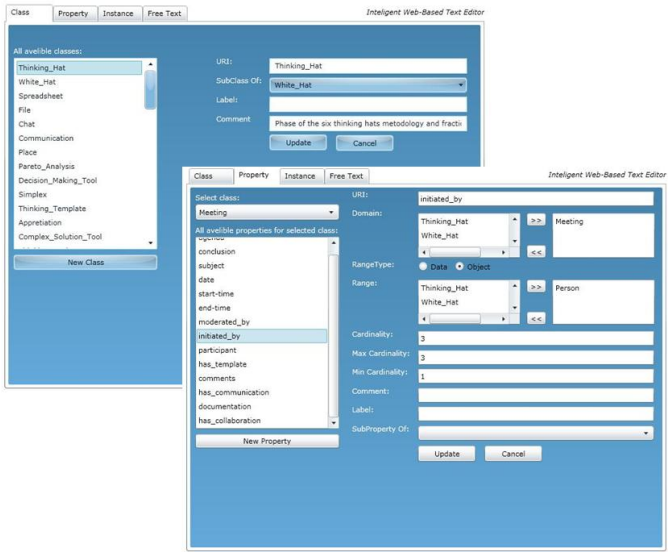

Figure 1: Class and property manipulation

At the upper - left side of the Figure 1 is shown the part of the user interface where the user can iterate through all the classes that are contained in the Jena Model. Here, the user can edit some of the classes, or, if he wants, he can create a new one and add it to the ontology. Creating new classes is easy and simple. Simply he have to type the *name* for the new class, choose the super class (inheritance is enabled) and some additional attributes like comment and label for newly create class. After this, the name of the new class appears in the list box that contains all the classes of the ontology. The second step is to add properties for this class. First of all the user have to choice one class from the combo box shown in Figure 1. After that, all properties whose domain is the class are listed in list box. By selecting one of them, the form (shown on the right side in Figure 1) is filled with data specific for the selected property. At this way, data can be edited and properties can be updated. Inserting new properties for an existing class is the same. Under the list, filled with all properties that might have one class, is positioned "New property" button. After clicking this button, the form for adding new property is reset and ready for new one. After filling the form, the new property will be added to the selected class from combo box that contains all classes.

The next tab is reserved for instance manipulation but the last is something different. The Instance tab offers the user to look all the instances that are inside the ontology. By selecting one of the offered classes, the user can see all the instances of that class. These instances are listed in list box show in Figure 2. The next step is selecting one of the instances. By doing that, the user can see which properties are describe the instance and he can see how their values. He can make changes and update the instance simply by clicking on Update button. The new updated instance is now inserted in the loaded ontology.

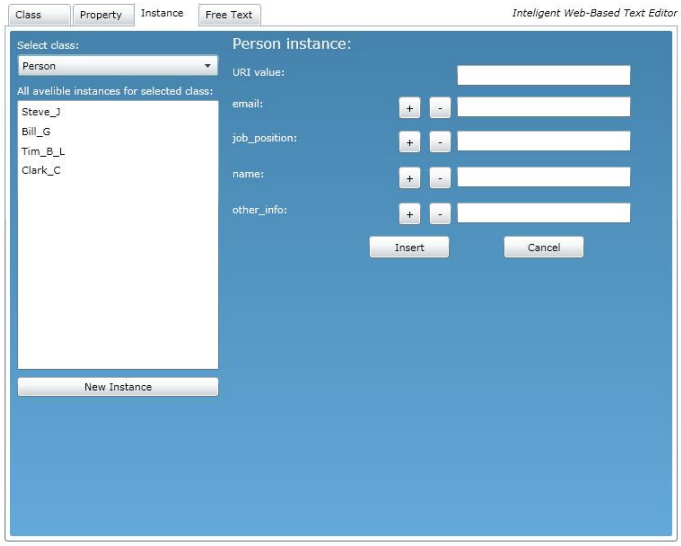

Figure 2: Working with instances

Inserting new instances of selected class is also available. Under the list containing all instances for selected class, is positioned "New instance" button. By clicking this button, the form on the right appears, containing all properties subscribed to selected class. After filling this form, new instance is created and added to list of all instances, available for selected class.

Best way to explain the last tab is by using an example. Let suppose that we have two classes: the Person class and

the City class. The Person class has property *is_from* (where the person lives) and the City class has property name (the name of the city). The ranges of the is_from property are instances of type City. We have created two instances of type Person: *Bill* who lives in London and Jon whose property is_from is not set.

Here comes the using of last tab. The user starts typing and he enter *Bill*. After he hit space in the auto complete box is shown is_from. The interface suggested the user that next he can enter the word is_from. After inserting is_from and hitting another space, the user interface knows that *Bill* is from "London", and in auto complete box is shown "London" (left side of Figure 3).

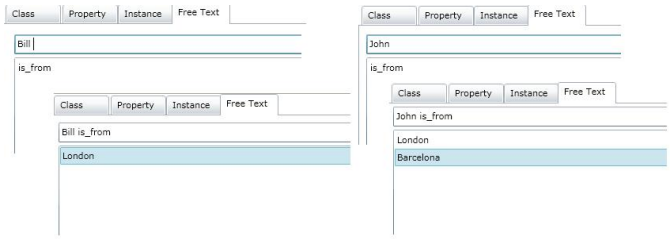

Figure 3: Manipulation with ontology using free text input

On the other hand if the user starts typing John and choose *is_from* from auto complete, after hitting another space because there is no information where John lives, in the auto complete box are shown all available cities (in our case London and Barcelona) and the user can choose one of them. This example is shown on Figure 3. By doing this, the user is guided by auto complete feature, which intelligently suggest and offer next words that might be added in process of linking and connecting data contained in the ontology.

Architecture and Technical Design

From an architectural point of view, this application has two main parts and a few subparts. First part is the knowledge management system (KMS), that makes persistence and manipulation with the ontology. Behind this knowledge management system is the Jena framework that is responsible for handling request that makes changes to the ontology. In addition, Jena takes care of making some reasoning based on the ontology, and due to this operation, new links and connections between the resources are made. Because this operation of reasoning is very expensive, in terms of time and resources, it is not done very often. The other thing that is more significant for our application are the web services that this KMS provides. Because our application is Silverlight application and KMS and Jena were Java applications, the need of web services was inevitable. Therefore, this KMS offers a set of few operations, significant for ontology and data manipulation.

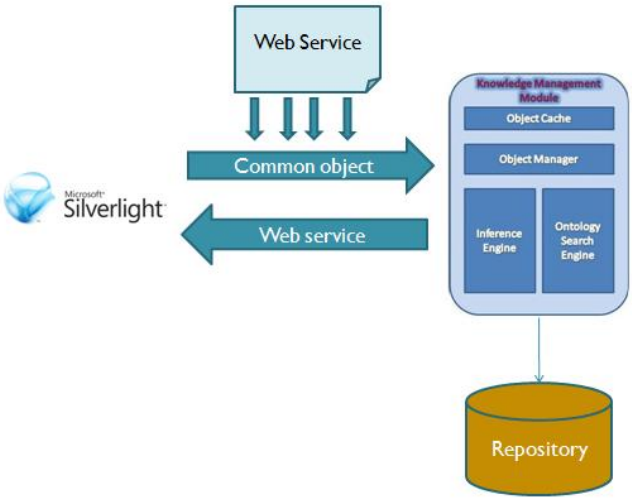

Figure 4: Application architecture and technical design

Because Microsoft Silverlight 2.0 [6] is technology which communicate with other applications only via Silverlight – enabled web services, a translation was necessary. Therefore, the data that were provided by KMS first were received by ASP.NET application [7], and then forwarded to the Silverlight application via Silverlight – enabled web services. Visually this architecture is represented on Figure 4. After the data is received, the application displays properly and performs operations over them. Depending on this operation, corresponding forms are filled, or assembled, and data is represented.

Additionally, because operations for retrieving classes and properties from ontology sometimes may take a while, caching of data is performed. At the beginning of the application, before the user interface is displayed, data for classes and its properties are imported in the application and cached. Therefore, operations where manipulations with classes and properties are needed, they are read from cache rather than making new call to web services. Updating data is made on both sides, on cached data and on data represented by the ontology.

Advantages and Disadvantages

We agree that every application has advantages and disadvantages. As first advantage of this application can be mention good way of caching data. That way the application runs faster, transparently executing operations for data manipulation. Next is the auto-complete feature that guides the user while he is writing natural sentences and tries to connect the resources. In addition, we have to mention that the forms for creating and editing classes, properties and instances are pretty simple and clear, so, the user can easily manipulate with the ontology. Another advantage is the transparent generation of the forms for entry and manipulation with the ontology resources. Thus we have a suitable form for every possible type of resource.

As a disadvantage of this application can be mentioned the inequality of the sequence by which the fields are created

when ontology instance is created. The order of the fields depends on how much properties have given instance, and its type. Listing all properties when creating an instance is also disadvantage of the user interface. There is a class that can contain several properties and filling all of them can be hard and painful. This can be resolved by using views and collections in some of the next improvements of the application.

Conclusion

With the work presented in this paper we achieve flexible system for data entry, that utilizes the benefits of the semantic web technologies. Due to the SOA [5] oriented design of this system, it is easier to maintain, and its parts can be reused in other systems, providing eater flexible data presentation and entry, or efficient manipulation with semantic data.

Future Work

Because the possibilities that offer the semantic web are huge, research in this area does not stop here. Our aim is to improve interface interactivity with the user, facilitating the way of entering date in process of building new ontology or editing existing one. Making heterogeneous user interface, which would be combination of forms and free typing, would make data entry very easy for the end user. Other good feature would be graphically display of the entire ontology, so the user can see the relationship of the data and its connection. We hope that this will be done in near future, and new editor with better performance and reasoning will be made.

References

[1] **RDF**, Resource Description Framework, http://www.w3.org/RDF

[2] **OWL,** Web Ontology Language**, http://www.w3.org/TR/owl-features/**

[3] Abraham Bernstein and Esther Kaufmann: **GINO - A Guided Input Natural Language Ontology Editor,** University of Zurich, Dynamic and Distributed Information Systems, Switzerland, 2007

[4] David R. Karger Karun Bakshi David Huynh Dennis Quan Vineet Sinha. **Haystack: A Customizable General-Purpose Information Management Tool for End Users of Semistructured Data**,MIT Laboratory for Computer Science, 2002

[5] Krafzig, D. and Banke, K. and Slama, D., **Enterprise SOA: Service-Oriented Architecture Best Practices (The Coad Series)**, Prentice Hall PTR Upper Saddle River, NJ, USA, 2004

[6] **Silverlight 2.0**, http://silverlight.net/

[7] **ASP.Net**, www.asp.net

[8] **Jena** – A Semantic Web Framework for Java, http://www.jena.sourceforge.net

View publication stats