A Tool for Searching the Semantic Web for Supplies Matching Demands

Zuzana Halanová, Pavol Návrat, Viera Rozinajová

Abstract: *We propose a model of searching semantic web that allows to incorporate data semantics into the searching process.. We concentrated to devising a searching module. Its inputs are two ontologies, describing supply and demand. Its output are supply offers that correspond to demand requests, ordered by relevance to the request. In the next step, we attempted to generalise our searching model to be able to cope with arbitrary ontologies.*

.

Key words: *Semantic Web, Ontology, Supply, Demand.*

INTRODUCTION

In the present the searching tools focus on searching the data on the web according to contents but without consideration to their meaning. Efforts on enrichment of the data on the Web by their meaning and interpretation of the meaning emerge in the recent years. Semantic web is an idea and an effort to create a web that would incorporate meaning. To completely realize the ideas of semantic web it is necessary to explore the contribution of it and justify the practical contribution to the wide public.

Our goal has been to design a searching tool, which would consider also the meaning of the data. The notion of meaning of the data could be interpreted in several ways in the semantic web. We have found it useful to focus also on representation of the data meaning by ontology.

SEMANTIC WEB

The semantic web is an extension of the contemporary web, where data have a defined value, allowing so a better co-operation of people and computers. The fundamentals of the semantic web is, that the machine „knows" the meaning of the data. In contemporary web, data is only presented. This is the reason why the computers in the semantic web are considerably more effective at automatic data processing.

The basic element of the semantic web are metadata. The metadata alone are insufficient. Necessary are also standards for syntactic coding and representation of semantics and semantic knowledge, in order for the machine to know how to perform the tasks effectively and in a meaningful way[1]. One of many services, which could be improved by means of semantic web is searching the data on the web.

ONTOLOGY

The meaning of the data can be recorded in the semantic web by an ontology. Ontology is a formal, explicit specification of the shared conceptualization [2]. This means, that ontology is a model of a segment of the real world, which is clearly defined and machine-processable. Ontology created within our project presents a prototype of a domain ontology. Domain ontology describes a specific concrete section, in our case the described part of the real world is the job market area.

The job market presents a place, where jobs are offered and requested. It is the place, where labour supply and labour demand exist at one place. Two types of users originate in this area. It is the person seeking job (job demand) and person seeking personnel (job supply). So both types of users can search in the job market, while one type of the user is trying to find a user of the other type. As far as consent in requirement and supply on

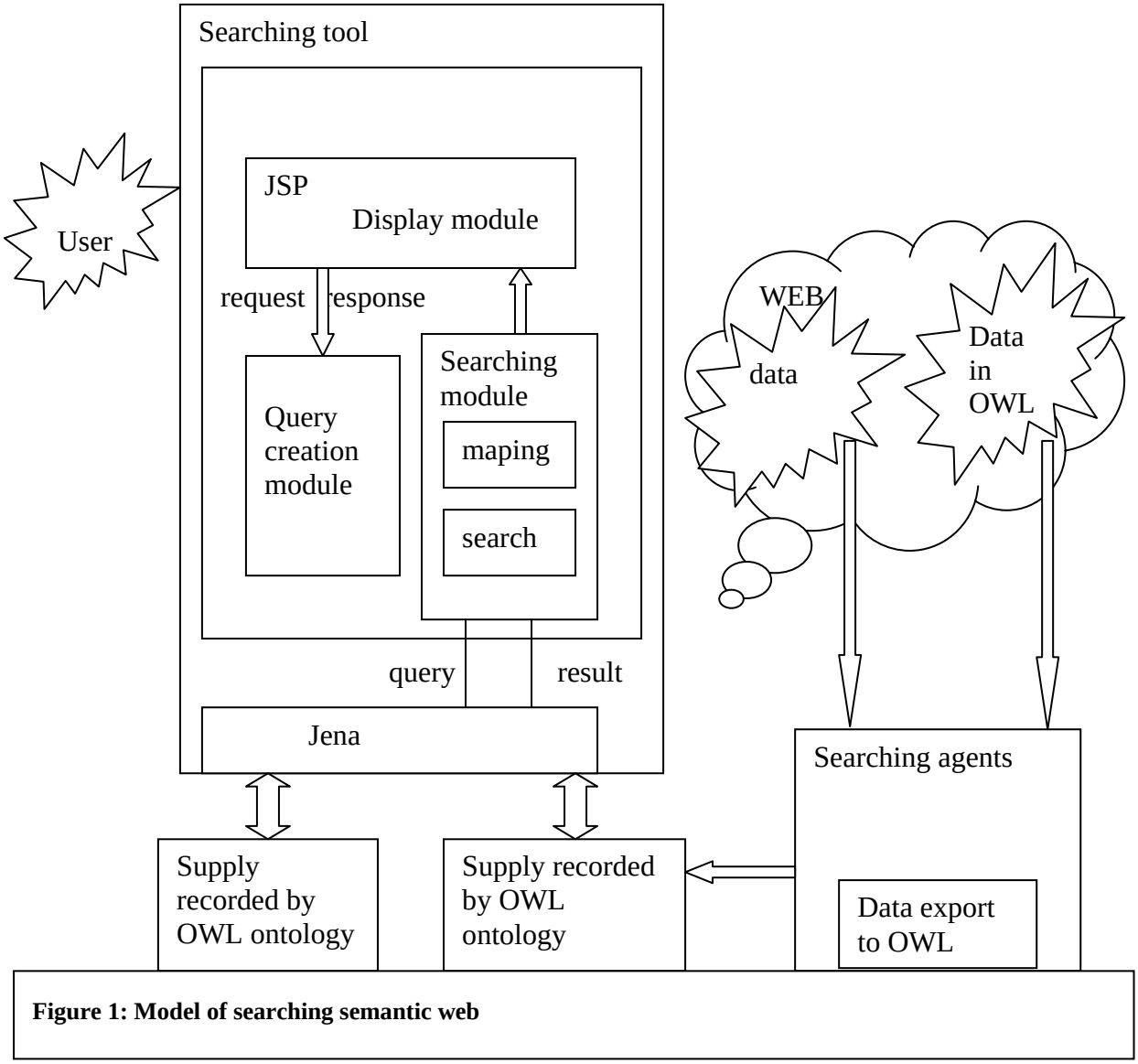

Figure 1: Model of searching semantic web

both sides arises, it comes to "pairing" of the persons seeking employment and the employers offering the job in the job market area.

We created ontology in the tool Protege [3] and with this tool we exported it into the format OWL. The language OWL is the latest standardized language for description of ontologies and that is why we decided to use it.

It is possible to find in the created ontology the description of the entities from the job market area.These are for example employment, business, supply on employment, accomplishments of the applicant, required experience and so on. In the proposition we divided ontology into a number of logical sections.They are representation of employment, representation of the business offering employment, representation of the offer of employment and representation of the reqiurements on the employee within the offer.Ontology proposed in this way makes it possible to add simply new entities or entity relationships without the need of changing the already created ontology. The created ontology was filled with data, which served as testing data for searching.

MODELLING SEMANTIC WEB SEARCH

The proposed model of searching on the semantic web is displayed in the picture. The model consists of several autonomous parts, which ensure some of the necessary activities of searching. They are compact parts of the searching tool and searching agents.The searching tool consists of several modules, that are interconnected by sending messages. The tool includes a display module, module for obtaining demand and a searching module. The tool works with data recorded by ontology. The data represent the searching criteria, i.e. demand and data to be searched,i.e. supply. The tool cooperates with the frame Jena [4], which ensures access to the data in ontologies.

The searching tool needs for its proceeding job offers written by OWL ontology. The tool does not require that all of the offers have to be written by the same ontology. The job offers can be obtained from the web through the use of searching agents, which run through the web and search relevant documents on a given troubleshooting area. The agent must be able to identify data regarding job offer area.

SEARCHING MODULE

The searching module consists of two sections. It is the section ensuring mapping and the section ensuring the searching itself. In the first step the tool is trying to find resemblance between ontologies of supply and demand, so it is the mapping of two ontologies. The tool identifies similar components in the given ontologies with a definite probability of concurrence. The searching is performed by data comparison of similar elements of the ontologies of supply and demand. The search results are arranged in order of relevance concerning demand. The obtained search results are sent in a message *response* to the display module, where they are arranged to an output and are displayed to the user.

Both sections of the searching module have multiple access to both ontologies. Frame Jena [4] was developed for working with ontologies. Communication between module and frame indicated in messages *query* and *result* is in the picture. The message *query* includes a question on finding the data in the ontology and the message *result* includes result of the search.

Input for this module are two ontologies, supply and demand. In this solution the possibility to reuse the given module is offered. However if we would use the knowledge of market ontology in mapping and searching , this module would be fixed on the given ontology. And so to allow reusing of the searching module we will process the ontology of supply and also demand as unknown. In this way it is possible to generalize searching on random domain section. A necessary condition of this approach is that both input ontologies are from the same domain section.

ONTOLOGY MAPPING

Before the searching itself it is necessary to find out, which attributes from the supply ontology match the individual attributes from the demand ontology. The requirement of mapping two ontologies arises. Mapping is a necessary condition of searching, that's why we encapuslated these two steps into one module.

Mapping is a process, where we try to find similarity between two ontologies. The idea of mapping consists of seeking similarities between individual elements of the ontologies. We compare the elements on the basis of their names and attributes.

The tool creates continuously a list of similar elements, where it also stores the probability of concurrence of the individual element pairs. The similarity of elements can be also determined on the basis of similarity of other elements, which relate to each other in a certain way. The equality probability of two elements is determined on the basis of the number of equal attributes of given elements and equality probability of given attributes.

The enumeration of equality probability of two elements in proccesed ontologies is suggested by following formula:

pzp = Σ pzv / number of attributes

where pzp is the equality probability of elements, pzv is the equality probability of attributes and number of attributes equals the number of attributes of the given element. We seek the similarity of elements of ontologies first among the classes themselves and then among the attributes.

When seeking concurrent attributes object-oriented attributes are tested at the beginning, then data attributes are processed and ultimately, object-oriented attributes with data are also compared.

When comparing attributes, the probability of concurrence of two attributes is computed according to the following formula:

pzv = (name + pzdom + pzran)/3

where pzv is the equality probability of attributes, name is equality of names of attributes, pzdom is the equality probability of attribute domains and pzran is the equality probability of range attributes (line of value). Parameter name is allowed to get only values 1, if the names consist or 0, if they don't. Parameter pzdom gets values always from interval <0,1>. Parameter pzran in case of testing data attributes gets only values 0 and 1, when testing object-oriented attributes gets values from interval <0, 1>. The distinction by testing various types of attributes originates in the style of determination of the concurrence probability range.

At the end of comparison of all the types of attributes we arrange the lists of similar attributes. In all the lists we keep every attribute only once and in that pair, that has the greatest concurrence probability of the given attributes. When mapping is finished, the tool has a list of similar elements, that is later used in searching.

SEARCHING

After determining similar elements through the use of mapping we can seek instances in supply according to criterias set in demand. After having found the suitable supplies it is necessary to designate the relevant classes in supply and classify the results according to relevance. Arrangement according to relevance is at the same time also arrangement according to probability of concurrence of found supplies with the assigned demand. Sending the searching results follows in message *response* to the display module.

Search is performed by gradual passing through all the class instances from both ontologies and compares these instances mutually. The instance is infered from a concrete class. If the instances resemble these classes, we can assume that they have similar attributes, these are the attributes of the given class. For each pair of similar classes we compare the instances of these two classes.

The concurrence probability is gradually calculated when browsing two instances. The instance attributes can get concrete values. The value of the instance attribute in demand is the value that we search for and value of the instance attribute in supply is the one we browse. So in searching we compare the values of attributes of individual instances. If the values of attributes coincide, the equality probability of the two given attributes is included into the equality probability of instances. Mathematically we express it by following formula:

pzi = aw (pzv*value_coincidence)/pvi

where pzi is the equality probability of the two compared instances, pzv is the equality probability of instance attributes, pvi is the number of attributes in an instance. Parameter value_coincidence can gain value 1, if the given attribute values consist and 0 if they don't. The equality probability of two attributes (parameter pzv) is obtained from the list of similar attributes, which was created within mapping. Parameter pzi can be zero in case

that for all the instance attributes the parameter value_coincidence is equal to zero. It means that the instances don't match any attribute value.
If pzi is different from zero after comparison of two instances, the given pair of instances is inserted into a list of similar ones together with the probability pzi. Searching is finished after comparison of instances of all similar classes and the results of searching is in a list of similar instances.

As from the point of view of the user the searching is finished when results are in output, it is necessary to process these similar instances further. It is a benefit for the user, when the research results are arranged according to the relevance concerning demand. If there are only instances of relevant class in the list, we can arrange them according to probability, which in this case determines the extent, by which the searched supply meets demand. Consequently it is possible to send the results to the display module.

SEARCHING AGENT

The supply for the searching tool represent data containing job offers recorded by ontology. On behalf of making the ontology accesible it is necessary at the beginning of the processing to provide the searching tool with URI of given ontology. The tool with supply ontology then works by means of Jena frame [4].

In the present we assume, that the data found on the web are presented more frequently in HTML format and in a minor extent written by ontologies. The task is to trace data on the web, which interest us, in our case job offers. In case of finding job offers written in OWL ontology it is sufficient to offer the URI of given ontology to the searching tool. In case of finding job offers written in the HTML format it is necessary to first inscribe these offers into OWL ontology and then provide them to the searching tool.
In both cases it is necessary to solve the resolution of relevant data by machine. The machine must be able to identify the data related to job offer. One of the promising posibillities of searching relevant documents on the web could be intelligent agents. These agents could have its own vocabulary of problem domain terms. They would search for the words from the vocabulary in web documents. Relevance of a given source could be assigned according to the density of word occurrence in regard to the processed domain. The agent would note down every relevant document by the means of an ontology. Here it would be necessary to consider, if for each problem domain there should be one fixed given ontology, or it would be better to choose the suitable ontology from some set. In this way the agents would impose semantics into not semantic data.
After processing the document, i.e.writing data into an ontology, this ontology would be provided to the searching tool. The tool would process the offer in an already described technique.

EVALUATION

Mapping is a necessary condition of searching. Corectness of searching depends directly on mapping results. We consider the search correct, if the entries (the data in the slots) of the found instances meet the entries of the search instances. The extent of similarity (equality) of entries expresses probability of equality of given instances.The more entries of instances match together, the bigger this number is.

The process of mapping is „sensitive" on the influence of external factors, which enter into the process by means of demand and supply ontologies. For example the difference in mapping results is significant, if we enter entry ontologies in a different sequence. The sequence of processing classes and attributes also has an effect on the results of mapping. The main objective by mapping ontologies is consecutive search of elements and relationships between them, while the search results influence further search of similarities. Hence logically if we change the search sequence, we also change the overall result of mapping.

There are also constraints on the entry ontology for the created process of mapping. If some ontology does not meet these constraints, the process of mapping and searching finishes with incorrect results.
In order to make the process of mapping resistant against these influences, it would be necessary to consider processing all the possible features of the language for the inscription ontologies, in our case OWL. However this would request a project of much greater extent. Thereby we would acquire a module, which is really capable to process an arbitrary ontology. However that is from a real perspective practically impossible and processing arbitrary ontologies will be always limited by some input conditions.
While processing the created demand and supply ontologies, the searching module works according to expectations. We assigned demand ontology as the first ontology, and supply ontology as the other. The search results are depicted in the example. On the basis of found similarities, searching has been carried out. There was a single class instance supply_searcher in the demand ontology and necessary instances of other classes. There were 8 job offers in the supply ontology. One supply matched demands in all aspects, while other offers matched demand only to a certain extent. The searching module determined the probability of matching supply and demand correctly, in accordance of an extent of matching found in data. The probability of a match of demand with one job offer in the searching results is 0.05. When displaying results in output we can designate the minimal value of match probability, that shall determine, which results we present to the user. On the list of similar instances we have all similar instances, also for example the instance of knowledge classes or Clanguage. However we are interested in class instances of Csupply.

DEMAND SUPPLY
offer_for searching_person Coffer Coincidence probability
---
offer_for searching_person_6 | trh_prace_Instance_10022 0.558333
offer_for searching_person_6 | trh_prace_Instance_25 0.308335
offer_for searching_person_6 | trh_prace_Instance_10027 0.258336
offer_for searching_person_6 | trh_prace_Instance_27 0.254165
offer_for searching_person_6 | trh_prace_Instance_20013 0.204166
offer_for searching_person_6 | trh_prace_Instance_26 0.204166
offer_for searching_person_6 | trh_prace_Instance_20012 0.154167
offer_for searching_person_6 – trh_prace_Instance_10026 0.05

Example:search results – similar class instances supply_searcher and Csupply and the probability of their consistency.
While testing the searching module we also examined input ontologies different from the available ontologies. Several of them contained also such OWL features, which were not recognized by our module. Despite that the module was able to process the examined ontologies too. However the unimplemented features were not taken into consideration while mapping and consequently searching. The positive feature of the tool is its ability to process these ontologies correctly, and thus capability to finish the processing without error statement.

CONCLUSIONS AND FUTURE WORK

By evaluating the project several questions arouse. The idea of creating a searching module, which can process arbitrary ontology written in the OWL language, enables its reusability. It is questionable, whether we are able to create such a general module, which in the process of mapping and searching would be able to take into consideration all features of the OWL language. Created module does not take into consideration all

possible OWL features, but is able to correctly,i.e. without error statements, process some ontologies which use these features. That suggests, that a way of creating a common searching module, seems to be very perspective.

The only domain independent element of our tool scheme is the searching module. Setting the criteria by the user, acquiring demand from the entered criteria, displaying the results of the search and the ontologies of demand and supply are domain dependent. Also the concrete agent searching for relevant documents in the world-wide web is domain dependent. For the creation of a complete searching tool it is necessary to implement domain dependent parts from the proposed model.In the concrete tool it is then sufficient to use the services of the searching module, which was in this project created. While using the services of the searching module , it is necessary to take into consideration constraints of processing an unknown ontology. This makes it possible to create a searching tool in arbitrary area using the created searching module.

ACKNOWLEDGEMENTS

This work was partially supported by the Slovak State Programme of Research and Development "Establishing of Information Society" under the contract No. 1025/04 and the Scientific Grant Agency of Republic of Slovakia, grant No. VG1/3102/06.

REFERENCES

[1] An, Y., Borgida, A., Mylopoulos, J.: Constructing Complex Semantic Mappings Between XML Data and Ontologies. In: Yolanda Gil, Enrico Motta, V. Richard Benjamins, Mark A. Musen (Eds.): The SemanticWeb - ISWC 2005, 4th International SemanticWeb Conference, ISWC 2005, Galway, LNCS 3729, Springer 2005, 6-20.

[2] Ehrig, M. - and Steffen Staab, S.: QOM – Quick Ontology Mapping. In: Sheila A. McIlraith, Dimitris Plexousakis, Frank van Harmelen (Eds.): The Semantic Web – ISWC 2004: Third International Semantic Web Conference, Hiroshima, LNCS 3298, Springer 2004, 683-697.

[3] Gasevic, D. and Hatala, M. Ontology mappings to improve learning resource search. *British Journal of Educational Technology*, Volume 37, Issue 3, Page 375-389, May 2006.

[4] Greenberg, J. – Sutton, S. – Campbell, D. Grant: Metadata: A Fundamental Component of the Semantic web. Bulletin of the American Society for Information Science and Technology, 2003, vol. 29, no. 4, s. 16-18.

[5] Kim, W., Choi D.W., Park, S.: Product Information Meta-search Framework for Electronic Commerce Through Ontology Mapping. In: A. Gómez-Pérez, J. Euzenat (Eds.): The Semantic Web: Research and Applications: Second European Semantic Web Conference, ESWC 2005, Heraklion, LNCS 3532, Springer 2005, 408 – 422.

[6] Stanford Medical Informatics: Protégé. http://protege.stanford.edu/ (2. 12. 2005)

[7] Jena – A Semantic Web Framework For Java. http://jena.sourceforge.net/ (2. 12. 2005)

ABOUT THE AUTHORS

Zuzana Halanová, Bc., Prof. Pavol Návrat, PhD., Viera Rozinajová, PhD., navrat@fiit.stuba.sk, from:

Institute of Informatics and Software Engineering, Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava, Slovakia.