Improving the accuracy of job search with semantic techniques

Malgorzata Mochol[1], Holger Wache[2], Lyndon Nixon[1]

[1] Freie Universität Berlin, Institut für Informatik
Takustr. 9, D-14195 Berlin, Germany
mochol,nixon@inf.fu-berlin.de
[2] Vrije Universiteit Amsterdam
Artificial Intelligence Department de Boelelaan 1081a, NL-1081HV
Amsterdam The Netherlands
E-mail: holger@cs.vu.nl

Abstract. In this paper we introduce a prototype job portal which uses semantically annotated job offers and applicants. In our opinion, using Semantic Web technologies substantially increase market transparency, lower transaction costs and speed up the procurement process. However adding semantics is not a panacea for everything. We identify some outstanding problems in job search using the system and outline how the technique of query approximation can be the basis for a solution. Through an Industry-Research co-operation we are extending the prototype with these semantic techniques to demonstrate a more accurate job search.

1 Introduction

Nowadays many business transactions are carried out via the Internet. Human resources management has also discovered the Internet as an effective communication medium. As reported in [7] 90% of human resource managers in Germany rated the Internet as an important recruitment channel. One reason for this high rate is that the Internet, in particular, reaches applicants who are young and highly qualified. Despite the fact that companies use more than one channel to publish their job openings, over half of all personnel recruitment is the result of online job postings [13]. Despite these achievements, the information flow in the online labour market is far from optimal.
As a result the German national project *Wissensnetze* (*Knowledge Nets*[3]) has worked together with German job portal provider WorldWideJobs GmbH [4] on developing an innovative semantic approach to searching for job offers or potential applicants on the Internet. In this paper we will present this approach, which has been demonstrated in a web-based prototype job portal. However, it is important to recognize that adding semantic annotation to the existing syntactic

[3] `http://wissensnetze.ag-nbi.de`
[4] `http://www.worldwidejobs.de/`

data is not a panacea for everything. On the other hand, the use of semantics based on formal logic models such as the RDF and OWL specifications within a human resource domain [5, 10] brings with it the possibility of new solutions to old problems, which would not be possible with pure syntactic data. In this paper, we introduce the approach of query approximation and show how it applies to the semantic job portal, further improving the results of job search. As a result we wish to demonstrate that while all aspects of a business problem will not be solved immediately by annotating semantically the existing business data, the introduction of semantics into business systems is the first step towards uncovering new IT solutions to typical business processes.
The rest of the paper is organized as follows: Section 2 gives a brief overview of a job procurement taking into account current (Sec. 2.1) as well as future (Semantic Web based) solutions (Sec. 2.2). Section 3 describes the semantic search approach used in the semantic job portal at the same time identifying some still outstanding problems. These issues are addressed in Section 4 which outlines how query approximation techniques can be the basis for the possible solution of such difficulties. We summarize the research on the application of query relaxation on the job procurement tasks with a brief status of the work and outlook considering future activities in Section 5.

2 Job Procurement: Old and New Ways

2.1 State of the art

A large number of online job portals have sprung up, dividing the online labour market into information islands and making it close to impossible for a job seeker to get an overview of all relevant open positions. On the other side, the strong market position of the job portals as the prime starting point for job seekers allows them to charge employers high fees for publishing open positions which causes that in spite of a large number of portals employers still publish their openings on a rather small number of portals in order to keep costs and administrative effort down. The hiring organizations simply assume that a job seeker will visit multiple portals while searching for open positions.
Furthermore, the German Federal Employment Office (BA), for example, launched the platform "virtual employment market" in December 2003. This initiative is an effort to increase transparency of the job market and to decrease the duration of job procurement. In spite of high investments, these goals have not been reached yet. Some of the problems of the BA portal are, on the one hand, the necessity for all participants of the virtual employment market to use the proprietary data exchange format issued by the BA and some defects in quality of data and query results on the other hand.
Alternatively, companies can publish job postings on their own website [11]. This way of publishing, however, makes it difficult for job portals to gather and integrate job postings into their database. Thus search engines such as Google or Yahoo! have become vital in the job search. Furthermore, dedicated search

engines, such as worldwidejobs.de[5], are entering into the market, allowing detailed queries as opposed to the keyword-based search of current search engines. The quality of search results depends not only on the search and index methods applied. Further influential factors include the processability of the used web technologies and the quality of the automatic interpretation of the company-specific terms occurring in the job descriptions. The deficiencies of a website's machine processability result from the inability of current web technologies, such as HTML, to semantically annotate the content of a given website. Therefore, computers can easily display the content of a HTML site, but they lack the ability to interpret the content properly.

2.2 Semantic-based solution

In our opinion, using Semantic Web technologies in the domain of online recruitment and skill management can cope with such problems and substantially increase market transparency, lower the transaction costs for employers and speed up the procurement process. For this reason, in the *Wissensnetze* project, which explores the potential of Semantic Web from a business and a technical viewpoint by examining the effects of the deployment of Semantic Web technologies for particular application scenarios and market sectors, we developed a job portal which is based on Semantic Web technologies. Every scenario includes a technological component which makes use of the prospected availability of semantic technologies in a perspective of several years and a deployment component assuming the availability of the required information in machine-readable form. The combination of these two projections allows us, on the one hand, to build e-business scenarios for analysis and experimentation and, on the other hand, to make statements about the implications of the new technology on the participants of the scenario in the current early stage of development.
In a Semantic Web-based recruitment scenario the data exchange between employers, applicants and job portals is based on a set of vocabularies which provide shared terms to describe occupations, industrial sectors and job skills [12]. In this context, the first step towards the realization of the Semantic Web e-Recruitment scenario was the creation of a *human resources ontology (HR-ontology)*. The ontology was intended to be used in a job portal, not only to allow for a uniform representation of job posting and job seeker profiles but also to support semantic matching (a technique that combines annotations using controlled vocabularies with background knowledge about a certain application domain) in job seeking and procurement tasks. Another important requirement was extending the Semantic Web-based job portal with user needs, considering this is already common practice in the industry. Accordingly to this specification we focused on how vocabularies can be derived from standards already in use within the recruitment domain and how the data integration infrastructure can be coupled with existing non-RDF human-resource systems.

[5] `http://www.wwj.de`

In the process of ontology building we first identified the sub-domains of the application setting (skills, types of professions, etc.) and several useful knowledge sources covering them (approx. 25)[3]. As candidate ontologies we selected some of the most relevant classifications in the area, deployed by federal agencies or statistic organizations: German Profession Reference Number Classification (BKZ), Standard Occupational Classification (SOC), German Classification of Industrial Sector (WZ2003), North American Industry Classification System (NAISC), German version of the Human Resources XML (HR-BA-XML) and Skill Ontology developed by the KOWIEN Project[18]. Since these knowledge sources were defined in different languages (English/German) we first generated (depending on the language) lists of concept names. Except for the KOWIEN ontology, additional ontological primitives were not supported by the candidate sources. In order to reduce the computation effort required to compare and merge similar concept names we identified the sources which had to be completely integrated to the target ontology. For the remaining sources we identified several thematic clusters for further similarity computations. For instance BKZ was directly integrated into the final ontology, while the KOWIEN skill ontology was subject of additional customization. To have an appropriate vocabulary for a core skill ontology we compiled a small conceptual vocabulary (15 concepts) from various job portals and job procurement Web sites and matched them against the comprehensive KOWIEN vocabulary. Next, the relationships extracted from KOWIEN and various job portals were evaluated by HR experts and inserted into the target skill sub-ontology. The resulting conceptual model was translated mostly manually to OWL (since except for KOWIEN the knowledge sources were not formalized using a Semantic Web representation language) [15]. More information regarding our HR ontology can be found in [16, 12]
The planned architecture for the Semantic Web-based job portal which uses our HR-ontology implies three basic roles [2] (cf. Fig.1) information providers, aggregators and consumers.

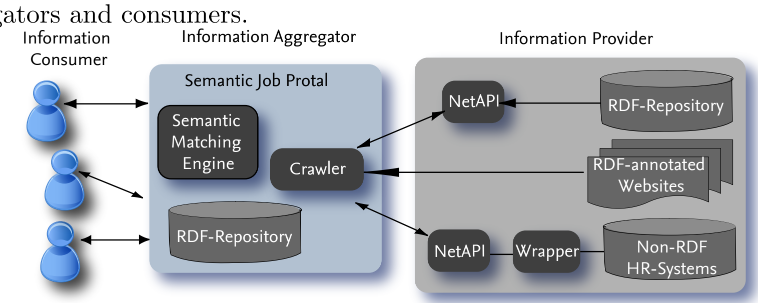

Fig. 1. Architecture of the Job Portal [2]

Information Providers who publish open positions and company background information in the RDF format using controlled vocabularies. They have two different approaches for publishing annotated job postings depending on their existing software infrastructure. If they use database standard software in the back-end, they can export related data directly into RDF using mapping tools like D2RQ [4]. If they do not use any enterprize software to manage open positions, they can annotate existing HTML versions of their postings using annotation tools and publish RDF version of their postings using, for example the RDF NetAPI or RAP.

Information Aggregators who crawl the published information and present it to the users in the desired format and level of aggregation. Combining job postings with background knowledge about the industrial sector allows the aggregators to offer semantic matching services. The usage of URIs as an employer identification mechanism allows the enrichment of postings with information about employers from different external information services.

Information Consumers who use one portal of their choice as a central access point to all published open positions and background information about employers, instead of collecting information fragments from different sites.

Having modelled the HR-ontology and prepared the RDF-Repository to store applicant profiles and job description, we developed the matching engine which as the core component of the system plays the crucial role in the procurement process (cf. Sec. 3).

3 Semantic job search

Semantic Matching is a technique which combines annotations using controlled vocabularies with background knowledge about a certain application domain. In our prototypical implementation, the domain specific knowledge is represented by concept hierarchies like skills, skill level classification, occupational classification, and a taxonomy of industrial sectors. Having this background knowledge of the recruitment domain (i.e. a formal definition of various concepts and specification of the relationships between these concepts) represented in a machine-understandable format allows us to compare job descriptions and applicant profiles based on their semantic similarity [17] instead of merely relying on the containment of keywords like most of the contemporary search engines do.
In our HR-scenario, our matching approach[6] utilizes metadata of job postings and candidate profiles and as the matching result, a ranked list of best candidates for a given job position (and vice versa) is generated. Inside both job postings and applicant profiles, we group pieces of information into "*thematic clusters*", e.g. information about skills, information regarding industry sector and occupation category, and finally job position details like salary information, travel requirement, etc. Each thematic cluster from a job posting is to be compared with the corresponding cluster from an applicant profile (and the other way round). The total similarity between a candidate profile and a job description is then calculated as the average of the cluster similarities. The *cluster similarity* itself is computed based on the similarities of semantically corresponding concepts from a job description and an applicant profile. The *taxonomic similarity* between two concepts is determined by the distance between them which reflects their respective positions in the concept hierarchy. Following this, the distance

[6] For further information about used matching framework SemMF see [14]

`d` between two given concepts in a hierarchy e.g. `Java` and `C` (cf. Fig. 2) represents the path from one concept to the other over the closest common parent. The semantic differences between upper level concepts are bigger than those between concepts on lower hierarchy levels (in other words, two general concepts like `ObjectOrientedLanguages` and `ImperativeProceduralLanguages` are less similar than two specialized like `C++` and `TURBOPASCAL`) and the distance between siblings is greater than the distance between parent and child (`d(Java,VisualBasic) > d(Java,PureObjectOrientedLanguages)`).

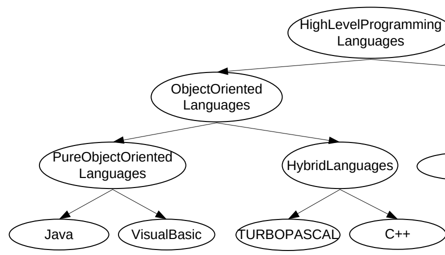

Fig. 2. Skills hierarchy

Since we also provide means for specifying *competence levels* (e.g. expert or beginner) in applicants' profiles as well as job postings we compare these levels in order to find the best match. Furthermore, our approach also gives employers the opportunity to specify the importance of different job requirements. The concept similarity is then justified by the indicated weight (i.e. the similarity between more important concepts) like the skills crucial for a given job position and will have greater influence on the similarity between a job position posting and an applicant profile.

3.1 Outstanding problems

In the first version of the *HR-prototype* we concentrated on the modelling of the human resource ontology and development of a matching approach for comparisons of applicant profiles and job openings with a focus on skills and occupations as well as industry sector descriptions. The further specification of the job portal contains the comparison of applicants and vacancies not only under consideration of skills and their levels but also professional experience of a job seeker in relation to the requirements of the hiring company. We want to be able to express not only the duration of particular experience (e.g. 3 years experience in Java programming) but also to deliver those job applications which maybe do not fit 100% to the defined requirements but are still acceptable for the employer (e.g. 3 years instead of 5 years industry experience). Furthermore, to verify the consistency of the job opening descriptions we also have to avoid the definition of nonsensical requirements like job postings which demand only very young yet highly qualified people (e.g. under 25 with at least 10 years work experience). Following this, we need an additional method which starts checking the data with the strongest possible query that is supposed to return the "best" answers satisfying most of the given conditions and then weaken the query if the returned result set is either empty or contains unsatisfactory results.

Since we have been working very close with one German job search engine we were able to define (in cooperation with the industry) several exemplary use cases which focuses on the definition of such requirements and issues. From the practical point of view the use-cases may represent the kind of queries which happen in the real world. However from the scientific point of view these use-cases are challenges to the techniques which we want to apply.

When implementing a (Semantic Web) job portal the requirements of the system depend on the meaningful use cases which are derived by the industrial project partner from its day to day business practices within the HR-domain. To clarify the still outstanding problems we will briefly present one such use case which (at first view) seems to be quite simple. However, if we look closer and try to represent the data in an ontology or satisfy the requirements in the semantic portal we will meet some difficulties, which at the same time show the complexity of such "simple" queries.

We are looking for a person which:

1. has experiences in CORBA, JavaScript, Java and Databases,
2. has worked for at least 5 years in industrial and 5 years in research projects,
3. should have experience as project or team manager,
4. should be not older then 25.

This example serves as a guideline and a thread in the rest of the article.

4 Query relaxation

It lays in the nature of this application that instead of returning many exact answers to a request only a few or zero answers will be returned. Normally the description of the job requirements are too specific to be matched. A few or zero answers would immediately disappoint users and they would not continue to use a job portal. Therefore the job portal has to return not only the exact matches; it also has to return the most similar matches to a query.

Since *Semantic Web reasoning* is founded on logics which are perfect in returning exact answers but are inappropriate in order to return similar answers, it must be adopted or extended for the usage in such cases. The obvious approach uses a similarity measure which calculates the similarity between the job posting and request(cf. Sec. 3). Such a function $f(p, r) \mapsto [0..1]$ directly provide a ranking between the results because answers which are more similar can be higher ranked. However, each of such function does not explain how the job posting and the request differ since it only returns a value like 0.78. Furthermore, the difference between answers can not explored, i.e. the difference with another answer with ranking value 0.79 is not obvious and is not explained.

In order to ensure that the job portal ranks certain answers higher than others — and can ensure that the system will respect our preferences during the ranking — similarity measures can be biased in that way that weights w_i are attached

to some parts of the calculation, i.e. $f(p, r) = \sum_{i=1}^{n} w_i * f_i(p, r)$. But again this give only the impression that the ranking function can be biased in the preferred way. Furthermore, it still does not explain the difference of one answer with the request or other answers. On the other hand, the user is directly able to specify how he wants to relax his request. The user may specify directly: "if nobody have 5 years industrial experience then I will also accept 3 years experience". Furthermore, the system can also explain how this set of returned answers is related to the original query, e.g. here comes now the answers not with 5 but with 3 years experiences (cf. Sec. 3.1).
In the following we describe an approach which uses *rewriting rules* to capture this knowledge explicitly and show how this knowledge is used to relax the original query into a set of *approximated queries*. We propose an approach for query rewriting based on conditional rewriting rules. This rewriting relaxes the over-constrained query based on rules in an order defined by some conditions. This has an advantage that we start with the *strongest possible query* that is supposed to return the "best" answers satisfying most of the conditions. If the returned result set is either empty or contains unsatisfactory results, the query is modified either by replacing or deleting further parts of the query, or in other words relaxed. The *relaxation* should be a continuous step by step, (semi-)automatic process, to provide a user with possibility to interrupt further relaxations.
In [6] the authors proposed a rule-based query rewriting framework for RDF queries independent of a particular query language. The framework is based on the notion of triple patterns[7] as the basic element of an RDF query and represents RDF queries in terms of three sets:

- triple patterns that must be matched by the result (mandatory patterns);
- triple patterns that may be matched by the results (optional triple patterns);
- conditions in terms of constraints on the possible assignment of variables in the query patterns.

Re-writings of such queries are described by transformation rules $Q \xrightarrow{R} Q'$ where Q the original and Q' the rewritten query generated by using R. Rewriting rules consist of three parts:

- a matching pattern represented by a RDF query in the sense of the description above;
- a replacement pattern also represented by an RDF query in the sense of the description above;
- a set of conditions in terms of special predicates that restrict the applicability of the rule by restricting possible assignments of variables in the matching and the replacement pattern.

A re-writing is now performed in the following way: If the predicates in the conditions are satisfied for some variable values in the matching and the replacement pattern and the matching pattern matches a given query Q in the sense that the

[7] RDF statements that may contain variables

mandatory and optional patterns as well as the conditions of the matching pattern are subsets of the corresponding parts of Q then these subsets are removed from Q and replaced by the corresponding parts of the replacement pattern.
To clarify the approach we take the example from the Section 3.1: someone who has experiences in CORBA, JavaScript, Java and Databases. Looking for such a person requires the system to translate this free text description into an instance retrieval problem[8]. The query must be translated into a concept expression. The retrieval process will return all job seekers which belong to that concept expression, i.e. fulfill all the requirement in the concept expression. The following OWL expression shows the concept expression for some person who has experience in some of CORBA, JavaScript, Java and Databases[9].

```
<owl:Class rdf:ID="Query">
  <rdfs:subClassOf>
    <owl:Class rdf:ID="Person"/>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:someValuesFrom>
        <owl:Class>
          <owl:intersectionOf rdf:parseType="Collection">
            <owl:Class rdf:about="CORBA"/>
            <owl:Class rdf:about="JavaScript"/>
            <owl:Class rdf:about="Java"/>
            <owl:Class rdf:about="Databases"/>
          </owl:intersectionOf>
        </owl:Class>
      </owl:someValuesFrom>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="hasExperience"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
  ...
</owl:Class>
```

In the following we give some examples for the rewriting rules which use the aforementioned example as a basis.
Making use of the predicates we can use generic rewriting rules that are guided by information from the ontology. The predicate `subsumed` for example is satisfied when X is more specific than Y. With the following rewriting rule we are able to consider the knowledge in the ontology.

[8] Users provide the appropriate description manually in the structured form over the web-interface.

[9] Originally we modelled these as nominals (enumerations like Week =Monday, Tuesday, ...). Nominals are instances and classes at the same time. However current DL systems have problems with nominals therefore we use classes in the current approach.

```
pattern(<owl:Class rdf:about="X"/>) ==>
 replace(<owl:Class rdf:about="Y"/>) && subsumed(X,Y).
```

Following the general rewriting rule a specific rule taking into account a required skill, e.g. Java, can be defined. The simplest approach relaxes some requirements in the applicant's experience, i.e. instead of `JAVA` the`PureObjectOrientedLanguages` or even the`ObjectOrientedLanguages` could be possible weakenings of the original query (cf. Fig. 2):[10]

```
pattern(<owl:Class rdf:about="Java"/>) ==>
 replace(<owl:Class rdf:about="PureObjectOrientedLanguages"/>) && true.
```

This means that if the term `<owl:Class rdf:about="Java"/>` appears in a query it can be replaced by:
`<owl:Class rdf:about="PureObjectOrientedLanguages"/>`. This rule is a very specific one as it only applies in one special case and does not make use of information stored in the experience ontology. In the same way some number restrictions can be applied. In our example the requirement that a person has five years experience in industrial projects is encoded with the help of the (artificial) class `FiveYearsOrMore`. This class represents all numbers representing years which are larger or equal to five. This class can be replaced by the class `TwoYearsOrMore` which obviously is more general (weaker) then the former. Furthermore we can restrict the replacement in that way that we only allow this for the restriction on property `hasDuration`. The corresponding rewriting rule look like:

```
pattern(<owl:Restriction>
         <owl:onProperty rdf:resource="#hasDuration"/>
          <owl:someValuesFrom>
           <owl:Class rdf:ID="FiveYearsOrMore"/>
          </owl:someValuesFrom>
         </owl:Restriction>)
 ==>
 replace(<owl:Restriction>
          <owl:onProperty rdf:resource="#hasDuration"/>
           <owl:someValuesFrom>
            <owl:Class rdf:ID="TwoYearsOrMore"/>
           </owl:someValuesFrom>
         </owl:Restriction>)
 && true.
```

The main problem of the rewriting approach to query relaxation is the definition of an appropriate control structure to determine in which order the individual rewriting rules are applied to generalize new queries. Different strategies can be applied to deal with the situation where multiple re-writings of a given query are possible. Example is a Divide and Conquer (i.e. Skylining) [8, 9] strategy:

[10] For the sake of readability the examples are simplified.

The best results of each possible combinations of re-writings is returned. In the current version of the system we have implemented a simple version of skylining. In particular, we interpret the problem of finding relaxed queries as a classical search problem. The search space (Fig. 3) is defined by the set Q of all possible queries. Each application of a rewriting rule R on a query Q is a possible action denoted as $Q \xrightarrow{R} Q'$. A query represents a goal state in the search space if it does have answers. In the current implementation we use breadth-first search for exploring this search space. Different from classical search, however, the method does not stop when a goal state is reached. Instead the results of the corresponding query are returned, goal state is removed from the set of states to be expanded and search is continued until there are no more states to be expanded. As each goal state represents the best solution to the relaxation problem with respect to a certain combination of re-writings, the goal states form a skyline for the rewriting problem and each of them is returned to the user together with the query answers.

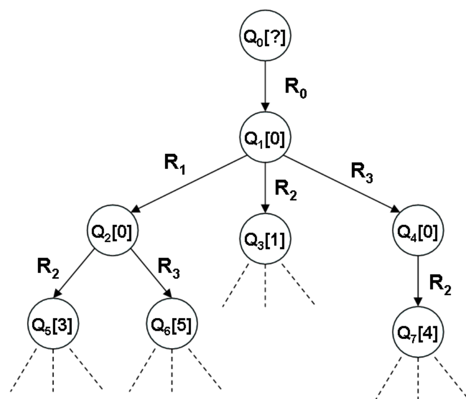

Fig. 3. Search space

The second difference to classical search is that we do not allow the same rule to be applied more than once with the same parameters in each branch of the search tree. The only kind of rules that can in principle be applied twice are rules that add something to the query[11]. Applying the same rule to extend the query twice leads to an unwanted duplication of conditions in the query that do not change the query result, but only increase the complexity of query answering.

5 Status and future work

The above mentioned co-operation between leading research in the area of semantic query approximation and the industry scenario of a semantic job portal has been enabled by the EU Network of Excellence KnowledgeWeb[12]. The aim of KnowledgeWeb is to enable transfer of semantic technologies from academia

[11] Rules that delete or replace parts of the query disable themselves by removing parts of the query they need to match against

[12] http://knowledgeweb.semanticweb.org/o2i

to industry. To achieve this, one of the first steps taken in the network was to collect industrial use cases where semantic technologies could form a potential solution, as well as derive from those use cases industry requirements for Semantic Web research. The results of this use case collection and analysis showed a real potential for the Semantic Web to solve real world business problems [1].
Currently, Knowledge Web is promoting the co-operation of leading Semantic Web researchers with selected industry partners with a real world business problem to solve. The co-operation between VU Amsterdam, Free University Berlin and WorldWideJobs GmbH is just one of these co-operations. In the co-operation on the semantic job portal, the following tasks have already been achieved:

- Ca. 10-20 good examples of queries which will include the characteristic of experience have been defined,
- The HR ontology has been extended in order to be able to model the property of experience,
- Ca. 250 job seeker and 250 job position instances have been added to the HR knowledge base which include the extended properties of the ontology and can be used to test semantic queries such as the one given in Sec. 3.1,
- An interface to the rule rewriting tool has been specified (which is more general than the DIG interface in that it is not limited to Description Logics),
- The first concrete technical details of rule rewriting (e.g. abstract query syntax) have been defined,
- A first implementation of the rule rewriting tool has been completed.

It is planned by early 2007 to have a working extended version of the semantic job portal which will support queries which cover the types of sample query mentioned in Sec. 3.1. A benchmark will be used to test the extended prototype against the original prototype which does not support query approximation. As a result, we hope to acquire a clear view of the benefits of applying query approximation to enable more relevant responses to over-constrained or inconsistent queries, which is currently not possible in the present prototype.
The aim of query approximation is to allow more robust and efficient query response from knowledge bases which can scale to real world enterprise size. Furthermore, this approach is useful in areas such as eRecruitment to loosen queries that are too specific in order to allow users to find best matches rather than simply receive no results at all. Extending the HR-prototype provides us with a real life example to test the value of the query approximation approach. On top of the extended prototype we plan to continue our research in testing the scalability of the approach, as well as techniques for modelling and generating rewriting rules.
Acknowledgement The work is supported by the EU Network of Excellence KnowledgeWeb (FP6-507482) and Wissensnetze Project which is a part of the InterVal-Berlin Research Centre for the Internet Economy funded by the German Ministry of Research.

References

1. L. Nixon A. Leger, F. Paulus and P. Shvaiko. Towards a successful transfer of knowledge-based technology to European Industry. In *Proc. of the 1st Workshop on Formal Ontologies Meet Industry (FOMI 2005)*, 2005.

2. C. Bizer, R. Heese, M. Mochol, R. Oldakowski, R. Tolksdorf, and R. Eckstein. The Impact of Semantic Web Technologies on Job Recruitment Processes. In *International Conference Wirtschaftsinformatik (WI'05)*, 2005.
3. C. Bizer, M. Mochol, and D. Westphal. Recruitment, report, April 2004.
4. C. Bizer and A. Seaborne. D2RQ - Treating Non-RDF Databases as Virtual RDF Graphs. In *Proc. of the 3rd International Semantic Web Conference*, 2004.
5. M. Bourse, M. Leclère, E. Morin, and F. Trichet. Human Resource Management and Semantic Web Technologies. In *Proc. of the 1st International Conference on Information Communication Technologies: from Theory to Applications*, 2004.
6. Peter Dolog, Heiner Stuckenschmidt, and Holger Wache. Robust query processing for personalized information access on the semantic web. In *7th International Conference on Flexible Query Answering Systems (FQAS 2006)*, number 4027 in LNCS/LNAI, Milan, Italy, June 2006. Springer.
7. T. Keim et al. Recruiting Trends 2005. Working Paper No. 2005-22. efinance Institut. Johann-Wolfgang-Goethe-Universität Frankfurt am Main., 2005.
8. Werner Kießling and Gerhard Köstler. Preference sql - design, implementation, experiences. In *In Proc. of 28th International Conference on Very Large Data Bases*, pages 990–1001. Morgan Kaufmann, 2002.
9. M. Lacroix and Pierre Lavency. Preferences; putting more knowledge into queries. In Peter M. Stocker, William Kent, and Peter Hammersley, editors, *VLDB'87, Proceedings of 13th International Conference on Very Large Data Bases, September 1-4, 1987, Brighton, England*, pages 217–225. Morgan Kaufmann, 1987.
10. T. Lau and Y. Sure. Introducing ontology-based skills management at a large insurance company. In *Proc. of the of the Modellierung 2002*, pages 123–134, 2002.
11. W. Mülder. Personalinformationssysteme - Entwicklungsstand, Funktionalität und Trends. *Wirtschaftsinformatik. Special Issue IT Personal*, 42:98–106, 2000.
12. M. Mochol and E. Paslaru Bontas. Practical Guidelines for Building Semantic eRecruitment Applications. In *International Conference on Knowledge Management, Special Track: Advanced Semantic Technologies (AST' 06)*, 2006.
13. Monster. Monster Deutschland and TMP Worldwide: Recruiting Trends 2004. In *2. Fachsymposium für Personalverantwortliche*. Institut für Wirtschaftsinformatik der Johann Wolfgang Goethe-Universität Frankfurt am Main, 2003.
14. R. Oldakowski and C. Bizer. SemMF: A Framework for Calculating Semantic Similarity of Objects Represented as RDF Graphs. In *Poster at the 4th International Semantic Web Conference (ISWC 2005)*, 2005.
15. E. Paslaru Bontas and M. Mochol. Towards a reuse-oriented methodology for ontology engineering. In *Proc. of 7th International Conference on Terminology and Knowledge Engineering (TKE 2005)*, 2005.
16. E. Paslaru Bontas, M. Mochol, and R. Tolksdorf. Case Studies on Ontology Reuse. In *Proc. of the 5th International Conference on Knowledge Management (iKnow05)*, 2005.
17. J. Poole and J.A. Campbell. A Novel Algorithm for Matching Conceptual and Related Graphs. *Conceptual Structures: Applications, Implementation and Theory*, 954:293–307, 1995.
18. F. Sowa, A. Bremen, and S. Apke. Entwicklung der Kompetenz-Ontologie für die Deutsche Montan Technologie GmbH, 2003.