An Infrastructure for Building Semantic Web Portals

Yuangui Lei, Vanessa Lopez, and Enrico Motta

Knowledge Media Institute (KMi), The Open University, Milton Keynes,
{y.lei, v.lopez, e.motta}@open.ac.uk

Abstract. In this paper, we present our KMi semantic web portal infrastructure, which supports two important tasks of semantic web portals, namely metadata extraction and data querying. Central to our infrastructure are three components: i) an automated metadata extraction tool, *ASDI*, which supports the extraction of high quality metadata from heterogeneous sources, ii) an ontology-driven question answering tool, *AquaLog*, which makes use of the domain specific ontology and the semantic metadata extracted by ASDI to answers questions in natural language format, and iii) a semantic search engine, which enhances traditional text-based searching by making use of the underlying ontologies and the extracted metadata. A semantic web portal application has been built, which illustrates the usage of this infrastructure.

1 Introduction

The Semantic Web [1] is the vision of the next generation of the World Wide Web, in which information is given well-defined meaning and thus becomes understandable and consumable not only for humans, but for computers as well. A number of semantic web portals have been developed through which semantic mark-ups can be gathered, stored and accessed in certain communities. Examples include MindSwap[1], OntoWeb[2], Knowledge Web[3], CS AKTive Space[4], Flink[5] and MuseumFinland[6]. These semantic web portals extend the concept of web portals by adding semantics to the contents or services, so that they can be seen as applications of the semantic web delivered in relatively small user communities.

One important task of these semantic web portals is to offer both end users and applications a seamless access to the knowledge contained in the underlying heterogeneous sources. As such, it is important to ensure that i) high quality knowledge (in terms of semantic metadata, i.e., data represented in terms of

[1] http://www.mindswap.org/
[2] http://www.ontoweb.org/
[3] http://knowledgeweb.semanticweb.org/
[4] http://triplestore.aktors.org/demo/AKTiveSpace/
[5] http://flink.semanticweb.org/index.jsp
[6] http://museosuomi.cs.helsinki.fi/

domain specific ontologies) is extracted from heterogeneous sources in an automated manner, and ii) comprehensive querying facilities are provided, enabling knowledge to be accessed as easily as possible.

Our overview of current semantic web portals reveals that they offer limited support for metadata extraction and data querying. In contrast with these, the system we present here, the KMi semantic web portal infrastructure, focuses on these issues. Central to our infrastructure are three components: i) *ASDI*, an automated semantic data acquisition tool, which supports the acquisition of high quality metadata from heterogeneous sources, ii) *AquaLog* [10], a portable ontology-driven question answering tool, which exploits available semantic markups to answer questions in natural language format, and iii) *a semantic search engine*, which enhances keyword searching by making use of the underlying domain knowledge (i.e. the ontologies and the extracted metadata).

The rest of the paper is organized as follows. We begin in section 2 by investigating how current semantic web portals approach the issues of metadata extraction and data querying. We then present an overview of the KMi semantic web portal infrastructure in section 3. Thereafter, we explain the core components of the infrastructure in sections 4, 5, and 6. In section 7, we present the application of our infrastructure and describe the experimental evaluations carried out in this application. Finally, in sections 8, we conclude our paper with a discussion of our results, the limitations and future work.

2 State of the art

In this section, we investigate how current semantic web portals address the two important issues described above namely metadata extraction and data querying. As we focus only on these two issues, other facilities (e.g., the generation of user interfaces, the support for ontology management) will be left out. We survey a representative sample of portals and tools without performing an exhaustive study of this research strand.

MindSwap, OntoWeb, and Knowledge Web are examples of research projects based semantic web portals. They rely on back-end semantic data repositories to describe the relevant data. Some portals (e.g. MindSwap and OntoWeb) provide tools to support metadata extraction. Such tools are however not directly deployed in these portals. Regarding data querying, most portals offer ontology-based searching facilities, which augment traditional information retrieval techniques with ontologies to support the querying of semantic entities.

The CS AKTive Space [12], the winner of the 2003 Semantic Web Challenge competition, gathers data automatically on a continuous basis for the UK Computer Science domain. Quality control related issues such as the problem of duplicate entities are only weakly addressed (for example, by heuristics based methods or using manual input). The portal offers several filtering facilities (including geographical filtering) to support information access. Regarding data

querying, it provides a querying language based interface, which allows users to specify queries using the specified languages (e.g., SPARQL [7]).

MuseumFinland [7] is the first semantic web portal that aggregates heterogeneous museum collections. The underlying metadata is extracted from distributed databases by means of mapping database schemas to the shared museum ontologies. The portal provides a view-based multi-facet searching facility, which makes use of museum category hierarchies to filter information. It also provides a keyword searching facility, which attempts to match the keyword with the available categories and then uses the category matches to filter information.

Flink [11], winner of the 2004 Semantic Web Challenge competition, extracts and aggregates online social networks in the research community of the semantic web. The data is aggregated in an RDF(S) repository and a set of domain-specific inference rules are used to ensure its quality. In particular, identity reasoning is performed to determine if different resources refer to the same individual (i.e., co-relation). FLink makes use of a range of visualization techniques to support information browsing. Data querying is however not supported.

In summary, most portals mentioned above support metadata extraction. While they do provide useful support in their specific problem domain, *their support for quality control is relatively weak*. Even though some co-relation and disambiguation mechanisms have been exploited (e.g., in CS AKTive Space and Flink), quality control has not been fully addressed. For example, the problems of duplicate or erroneous entities have not been addressed in any of the approaches mentioned above. Such problems may significantly decrease the quality of the acquired semantic data.

Regarding data querying, some portals (e.g. CS AKTive Space, MuseumFinland, and FLink) provide comprehensive support for information clustering and filtering. Some (e.g. MindSwap, MuseumFinland) provide ontology-based searching facilities. While these facilities do provide support for users to access the underlying information, *they typically suffer from the problem of knowledge overhead*, which is requiring users be equipped with extensive knowledge of the underlying ontologies or the specified query language in order to be able to i) formulate queries by means of filling out forms with ontology jargons (e.g., in OntoWeb) or specifying queries (e.g., in CS AKTive Space), ii) to understand the querying result (e.g., in MindSwap), or iii) to reach the data they are interested in (e.g., in MuseumFinland, FLink). Users are not able to pose questions in their own terms.

Another issue associated with data querying is that *the keyword-based searching support is relatively weak*. The benefits of the availability of semantic markups have not yet been fully exploited. Indeed, with a partial exception of MuseumFinland (which matches keywords against museum categories), no efforts have been made to try to understand the meaning of the queries. Furthermore, the techniques used are primarily focused on the enabling of search for semantic entities (e.g., MindSwap, OntoWeb). The underlying semantic relations of metadata have not been used to support the finding of other relevant information. For example,

[7] http://www.w3.org/TR/rdf-sparql-query/

when searching "phd student", we often get a list of instances of the class *phd student*. We may however be more interested in the news stories, the academic projects, the publications that phd students are involved in, rather than the instances themselves. The search engines will fail if such information is not directly defined in the class. This is because the complex relations of the semantic metadata have not been exploited.

3 An overview of the KMi semantic web portal infrastructure

Figure 1 shows the five layered architecture of our KMi semantic web portal infrastructure, which contains a source data layer, an extraction layer, a semantic data layer, a semantic service layer, and a presentation layer.

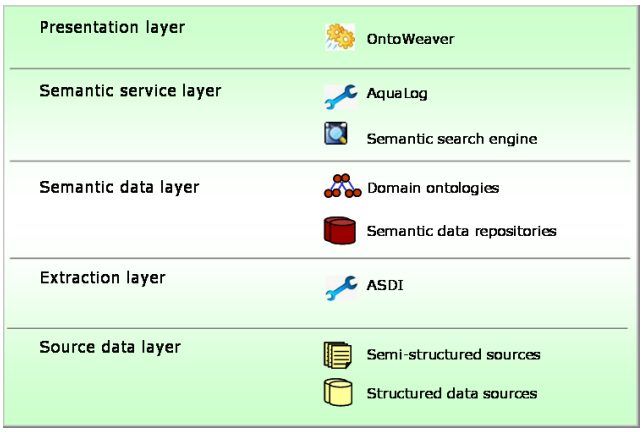

Fig. 1. An overview of the KMi semantic web portal infrastructure.

The source data layer comprises the collection of all available data sources such as semi-structured textual documents (e.g. web pages) or structured data in the form of XML feeds, databases, and knowledge bases.

The extraction layer is responsible for the extraction of high quality semantic data from the source data layer. The core component is the metadata acquisition tool ASDI. As will be described in section 4, ASDI provides several means to ensure the quality of the extracted data.

The semantic data layer contains a number of domain ontologies and semantic data repositories which store metadata extracted from the source data layer by the underlying extraction layer.

The semantic service layer provides semantic services over the semantic data repositories for the target semantic web portals. Data querying is seen as one of the important services, which enables knowledge to be easily accessed.

Central to this layer are two components. One is AquaLog, which takes questions in natural language format and an ontology as input and presents precise answers to users. As will be explained in section 5, AquaLog makes use of natural language processing technologies and the semantic representation of the extracted knowledge to achieve the task of question answering. The other component is the semantic search engine, which enhances the performance of traditional keyword search by augmenting current semantic search techniques with complex semantic relations extracted from the underlying heterogeneous sources. This component will be described in section 6.

The presentation layer supports the generation of user interfaces for semantic web portals. The KMi semantic web portal infrastructure relies on OntoWeaver-S [9] to support i) the aggregation of data from the semantic service layer and the semantic data layer and ii) the generation of dynamic web pages. OntoWeaver-S offers high level support for the design of data aggregation templates, which describe how to retrieve data and organize data content. As the generation of user interfaces is out of the scope of this paper, we will not go through the details of OntoWeaver-S.

4 The extraction of high quality metadata

To ensure high quality, we identify three generic tasks that are related to metadata extraction and which should be supported in semantic web portals. They include i) extracting information in an automatic and adaptive manner, so that on one hand the process can be easily repeated periodically in order to keep the knowledge updated and on the other hand different meanings of a given term can be captured in different context; ii) ensuring that the derived metadata is free of common errors; and iii) updating the semantic metadata as new information becomes available.

In ASDI we provide support for all these quality insurance related tasks. Figure 2 shows its architecture. ASDI relies on an *automatic and adaptive information extraction tool*, which marks-up textual sources, *a semantic transformation engine*, which converts data from source representations into the specified domain ontology according to the transformation instructions specified in *a mapping ontology*, and *a verification engine*, which checks the quality of the previously generated semantic data entries. These components will be explained in the following subsections.

4.1 Information extraction

To address the issue of *adaptive information extraction*, we use ESpotter [13], a named entity recognition (NER) system that provides an adaptive service. ESpotter accepts the URL of a textual document as input and produces a list of the named entities mentioned in that text. The adaptability is realized by means of domain ontologies and a repository of lexicon entries. For example, in

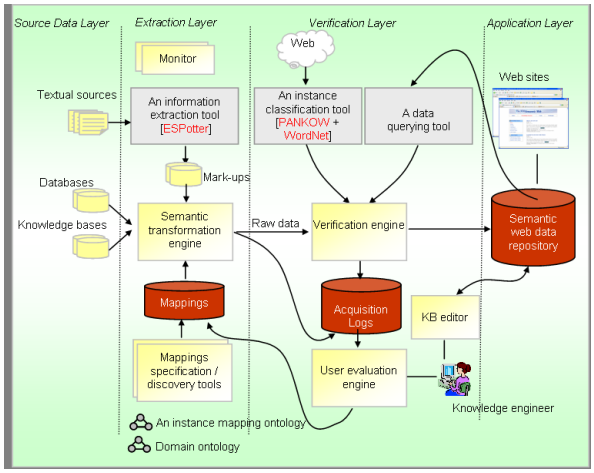

Fig. 2. The architecture of the metadata extraction tool ASDI.

the context of the KMi domain, ESpotter is able to mark the term "Magpie" as a project, while in other domains it marks it as a bird.

For the purpose of converting the extracted data to the specified domain ontology (i.e., the ontology that should be used by the final applications), an instance mapping ontology (see details in [8]) has been developed, which supports i) the generation of rich semantic relations along with semantic data entries, and ii) the specification of domain specific knowledge (i.e. lexicons). The lexicons are later used by the verification process. A semantic transformation engine is prototyped, which accepts structured sources and transformation instructions as input and produces semantic data entries.

To ensure that the acquired data stays *up to date*, a set of monitoring services detect and capture changes made in the underlying data sources and initiate the whole extraction process again. This ensures a sustainable and maintenance-free operation of the overall architecture.

4.2 Information verification

The goal of the verification engine is to check that each entity has been extracted correctly by the extraction components.The verification process consists of three increasingly complex steps as depicted in Figure 3. These steps employ several semantic web tools and a set of resources to complete their tasks.

Step1: Checking the internal lexicon library. The lexicon library maintains domain specific lexicons (e.g., abbreviations) and records the mappings between strings and instance names. One lexicon mapping example in the KMi semantic web portal is that the string "ou" corresponds to the instance *the-open-university* entity that has been defined in one of the domain specific ontologies. The verification engine will consider any appearances of this abbreviation as referring to the corresponding entity.

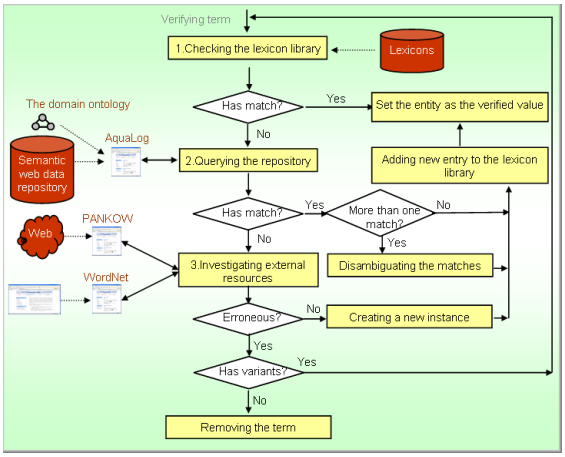

Fig. 3. The overall algorithm of the data verification engine.

The lexicon library is initialized by lexicons specified through the mapping instruction and expands as the verification process goes on. By using the lexicon library, the verification engine is able to i) exploit domain specific lexicons to avoid domain specific noisy data and ii) avoid repeating the verification of the same entity thus making the process more efficient.

Step2: Querying the semantic web data repository. This step uses an ontology-based data querying engine, to query the already acquired semantic web data (which is assumed to be correct, i.e. trusted) and to solve obvious typos and minor errors in the data. This step contains a disambiguation mechanism, whose role is to dereference ambiguous entities (e.g., whether the term "star wars" refers to the Lucas' movie or President Reagan's military programme).

The data querying engine employs a number of string matching algorithms to deal with obvious typos and minor errors in the data. For example, in a news story a student called *Dnyanesh Rajapathak* is mentioned. The student name is however misspelled as it should be *Dnyanesh Rajpathak*. While the student name is successfully marked up and integrated, the misspelling problem is carried into the portal as well. With support from the data querying engine, this problem is corrected by the verification engine. It queries the knowledge base for all entities of type *Student* and discovers that the difference between the name of the verified instance (i.e., *Dnyanesh Rajapathak*) and that of one of the students (i.e, *Dnyanesh Rajpathak*) is minimal (they only differ by one letter). Therefore, the engine returns the correct name of the student as a result of the verification. Note that this mechanism has its downfall when similarly named entities denote different real life objects.

If there is a single match, the verification process ends. However, when more matches exist, contextual information is exploited to address the ambiguities. The verification engine exploits the semantic relations between other entities appearing in the same piece of text (e.g. the news story) and the matches as the

contextual information. For example, when verifying the person entity *Victoria*, two matches are found: *Victoria-Uren* and *Victoria-Wilson*. To decide which one is the appropriate match, the verification engine looks up other entities referenced in the same story and checks whether they have any relation with any of the matches in the knowledge base. In this example, the *AKT* project is mentioned in the same story, and the match *Victoria-Uren* has a relation (i.e., *has-project-member*) with the project. Hence, the appropriate match is more likely to be *Victoria-Uren* than *Victoria-Wilson*.

Step3: Investigating external resources. If the second step fails, external resources such as the Web are investigated to identify whether the entity is erroneous, which should be removed, or correct but new to the system. For this purpose, an instance classification tool is developed, which makes use of PANKOW [2] and WordNet [4], to determine the appropriate classification of the verified entity. Now let us explain the mechanism by using the process of verifying the entity IBM as an example.

Step 3.1. The PANKOW service is used to classify the string *IBM*. PANKOW employs an unsupervised, pattern-based approach on Web data to categorize the string and produces a set of possible classifications along with ranking values. If PANKOW cannot get any result, the term is treated as erroneous but still can be partially correct. Thus, its variants are investigated one by one until classifications can be drawn. For example, the variants of the term "BBC news" are the term "BBC" and the term "news". If PANKOW returns any results, the classifications with the highest ranking are picked up. In this example, the term "company" has the highest ranking.

Step 3.2. Next the algorithm uses WordNet to compare the similarity between the type of the verified entity as proposed (i.e., "organization") and an alternative type for the entity as returned by PANKOW (i.e.,"company"). The algorithm here only checks whether they are synonyms. If they are (which is the case of the example), it is concluded that the verified entity is classified correctly. Thus, a new instance (*IBM* of type *Organization*) needs to be created and added to the repository. Otherwise, other major concepts of the domain ontology are compared to the Web-endorsed type (i.e.,"company") in an effort to find a proper classification for the entity in the domain ontology. If such classification is found, it is concluded that the verified entity was wrongly classified. Otherwise, it can be safely concluded that the verified entity is erroneous.

5 Question answering as data querying

In the context of semantic web portals, for a querying facility to be really useful, users have to be able to pose questions in their own terms, without having to know about the vocabulary or structure of the ontology or having to master a special query language. Building the AquaLog system has allowed us to address this problem.

AquaLog exploits the power of ontologies as a model of knowledge and the availability of up-to-date semantic markups extracted by ASDI to give precise,

focused answers rather than retrieving possible documents or pre-written paragraph of text. In particular, these semantic markups facilitate queries where multiple pieces of information (that may come from different sources) need to be inferred and combined together. For instance, when we ask a query such as "what is the homepage of Peter who has an interest on the Semantic Web?", we get the precise answer. Behind the scene, AquaLog is not only able to correctly understand the question but also competent to disambiguate multiple matches of the term *Peter* and give back the correct match by consulting the ontology and the available metadata.

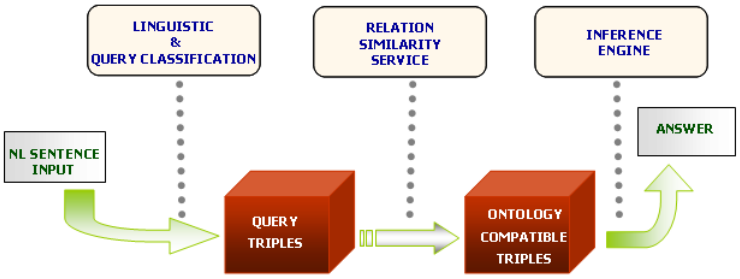

Fig. 4. An overview of the AquaLog architecture.

An important feature of AquaLog is its portability with respect to the ontologies. Figure 4 shows an overview of the AquaLog architecture. It relies on *a linguistic component*, which parses the natural language (NL) queries into a set of linguistic triples, *a relation similarity service* (RSS), which makes use of the underlying ontology and the available metadata to interpret the linguistic triples as ontological triples, and *an answer engine*, which infers answers from the derived ontological triples. To illustrate the question answering mechanism, we use the question *"what are the planet news in KMi related to akt?"* as an example. The process takes the following steps:

Step1: Parsing a NL query into linguistic querying triples. The linguistic component uses the GATE [3] infrastructure to annotate the input query and parses the query into a set of linguistic querying triples. At this stage the analysis is domain independent. It is completely based upon linguistic technologies. The example described above is parsed into two linguistic triples *(which is, planet news, KMi)* and *(which is, related to, akt)*.

Step2: Interpreting linguistic querying triples as ontology-compliant triples. In this step, the relation similarity service (RSS) disambiguates and interprets the linguistic querying triples, by making use of i) string metric algorithms, ii) lexical resources such as WordNet, and iii) the domain specific ontology. It produces ontology-compliant triples as results.

For the first linguistic triple (i.e. (which is, planet news, KMi)) derived from the example query, the RSS component matches the term *planet news* to the concept *kmi-planet-news-item* in the available metadata repository. It also identifies the term *KMi* as the instance *Knowledge-Media-Institute-at-the-Open-University* which is actually an instance of the concept *research-institute*.

Now, the problem becomes finding a relation which links the class *kmi-planet-news-item* to the class *research-institute* or viceversa. In order to find a relation all the subclasses of the non-ground generic term, kmi-planet-news-item, need to be considered. Thanks to ontology inference all the relations of the superclasses of the subject concept of the relation (e.g. "kmi-planet-news-item" for direct relations or "research institute" for inverse relations) are also its relations. The relations found include "owned-by", "has-author", and "mentions-organization". The interpreted ontology triples thus include *(kmi-planet-news-item owned-by research-institute)*, *(kmi-planet-news-item has-author research-institute)* and *(kmi-planet-news-item mentions-organization research-institute)*. Likewise, the second linguistic triple (i.e. *(which is, related to, akt)*) is linked to the first triple through the non-ground term *kmi-planet-news-item* and processed as ontology triples (*kmi-planet-news-item, mentions-project, akt)* and *(kmi-planet-news-item, has-publications, akt)*

Since the universe of discourse is determined by the particular ontology used, there will be a number of discrepancies between the NL questions and the set of relations recognized in the ontology. External resources like WordNet help in mapping unknown terms by giving a set of synonyms. However, in quite a few cases, such as in the two triples of our example, such resources are not enough to disambiguate between possible ontology relations due to the user or ontology specific jargon. To overcome this problem, AquaLog includes a learning mechanism, which ensures that, for a given ontology and a specific user community, its performance improves over time, as the users can easily give feedback and allow AquaLog to learn novel associations between the relations used by users, which are expressed in natural language, and the internal structure of the ontology.

Step3: Inferring answers from ontology-compliant triples. The answer engine looks through the available metadata to find data entries which satisfy the derived ontology triples and produces answers accordingly. As shown in figure 5, the answer of our query example is a list of instances of the class *kmi-planet-news-item* that mentions the project *akt*, in the context of the KMi semantic web portal, an application of our infrastructure, which will be explained in section 7. The user can navigate through each news entry extracted by the ASDI tool. Figure 5 also illustrates the derived linguistic triples and ontology triples.

6 Semantic search

Keyword search is often an appealing facility, as it provides a simple way of querying information. The role of *semantic* search is to make use of the underlying ontologies and metadata available in semantic web portals to provide

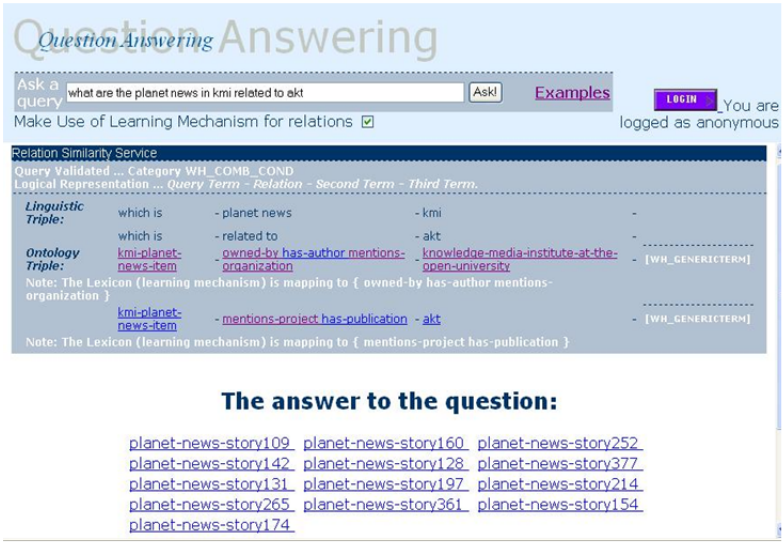

Fig. 5. An AquaLog running example.

better performance for keyword searching. In particular, the semantic search facility developed in our infrastructure extends current semantic search technologies[5, 6] (which are primarily centered around enabling search for semantic entities) by augmenting the search mechanisms with complex semantic relations of data resources which have been made available either by manual annotation or automatic/semi-automatic extraction in the context of semantic web portals.

To illustrate the semantic searching facility, we use the searching of news stories about phd students as an example. With traditional keyword searching technologies, we often get news entries in which the string "phd students" appears. Those entries which mention the names of the involved phd students but do not use the term "phd students" directly will be missed out. Such news however are often the ones that we are really interested in. In the context of semantic web portals where semantics of the domain knowledge are available, the semantic meaning of the keyword (which is a general concept in the example of phd students) can be figured out. Furthermore, the underlying semantic relations of metadata can be exploited to support the retrieving of news entries which are closely related to the semantic meaning of the keyword. Thus, the search performance can be significantly improved by expanding the query with instances and relations.

The search engine accepts a keyword and a search subject (e.g., news, projects, publications) as input and gives back search results which are ranked according to their relations to the keyword. The search subject is often domain dependent and can be predefined in specific problem domain. In customized portals, the search subjects can be extracted from user profiles. In the example described

above, the search subject is news stories. The search engine follows four major steps to achieve its task:

Step1: Making sense of the keyword. From the semantic meaning point of view, the keyword may mean i) general concepts (e.g., the keyword "phd students" which matches the concept *phd-student*), ii) instances entities (e.g., the keyword "enrico" which matches the instance *enrico-motta*, or iii) literals of certain instances (e.g., the keyword "chief scientist" which matches the values of the instance *marc-eisenstadt*). The task of this step is to find out into which category the keyword falls, by employing string matching mechanisms to match the keyword against ontology concepts, instances, and literals (i.e., non semantic entities, e.g., string values) step by step. The output of this step is the entity matches and the weight of the string similarity.

Step2: Getting related instances. When the keyword falls into the first category described above, the related instances are the instances of the matched concepts. In the second category case, the related instances are the matched instances. In the final category case, the related instances are the instances which have value as the matched literals. For example, when querying for the keyword "chief scientist", the matched instance is *marc-eisenstadt* whose value of the property *job-title* matches the keyword.

Step3: Getting search results. This step is to find those instances of the specified subject (e.g. news stories) which have explicit or (close) implicit relations with the matched instances. By explicit relations we mean the instances that are directly associated together in explicitly specified triples. For example, for instances i_1 and i_2, their explicit relations are specified in triples (i_1 p i_2) or (i_2 p i_1), where the entity p represents relations between them.

Implicit relations are the ones which can be derived from the explicit triples. Mediators exist in between, which bridge the relations. The more mediators exist in between, the weaker the derived relation is. For the sake of simplicity, we only consider the closest implicit relations, in which there is only one mediator in between. For instances i_1 and i_2, such implicit relations can be be represented as the following triples : i) (i_1 p_1 m), (m p_2 i_2); ii) (i_2 p_2 m), (m p_1 i_1); iii) (i_1 p_1 m), (i_2 p_2 m); and iv) (m p_1 i_1), (m p_2 i_2). In these relations, the semantic entity m acts as the mediated instance; the predicates p_1 and p_2 act as the mediated relations.

Step4: Ranking results. In this step, the search results are ranked according to their closeness to the keyword. We take into account three factors, including *string similarity*, *domain context*, and *semantic relation closeness*. The domain context weight applies to non-exact matches, which helps deciding the closeness of the instance matches to the keyword from the specific domain point of view. For example, with the keyword "enric", is the user more likely to mean the person *enrico-motta* than the project *enrich*? The domain context weight of a matched instance m_x is calculated as $\frac{Pm_x}{\sum_{i=1}^{n} Pm_i}$, where Pm_x denotes the count of the matched instance m_x serving as value of other instances in the metadata repository; and m_1, m_2, ..., and m_n represent all of the non-extact matches.

The semantic relation closeness describes how close the semantic relations are between a search result and the matched instances. The way of calculating it is to count all the relations a search result has with all of the matched instances. For this purpose, we give the explicit relations the weight 1.0, and the derived ones 0.5.

For the sake of experiments, we give each of the three factors described above (namely string similiary, domain context and semantic relation closeness) the same confidence. The confidence however can be easily changed to reflect the importance of certain factors. We applied the semantic search facility in the application of the KMi semantic web portal, which will be presented in the following section.

7 The KMi semantic web portal: an application

We have designed and tested our infrastructure in the context of building a semantic Web portal for our lab, the Knowledge Media Institute (KMi) at the UK's Open University, which provides integrated access to various aspects of the academic life of our lab[8].

7.1 Metadata extraction

The KMi semantic web portal has been built and running for several months generating and maintaining semantic data from the underlying sources in an automated way. The relevant data is spread in several different data sources such as departmental databases, knowledge bases and HTML pages. In particular, KMi has an electronic newsletter[9], which now contains an archive of several hundreds of news items, describing events of significance to the KMi members. Furthermore, related events are continuously reported in the newsletter and added to the news archive.

An experimental evaluation has been carried out to assess the performance of the metadata extraction in the context of the KMi semantic web portal, by comparing the automatically extracted data to the manual annotations in terms of *recall, precision* and *f-measure*, where recall is the proportion of all possible correct annotations that were found by the system with respect to the ones that can be in principle extracted from the source text, precision is the proportion of the extracted annotations that were found to be correct, and f-measure assesses the overall performance by treating recall and precision equally. The task is to recognize entities (including *people*, *projects* and *organizations*) mentioned in the randomly chosen news stories.

To illustrate the important role of the verification engine, the performance of ESpotter (which is the information extraction tool used in ASDI) is introduced in the comparison, which shows the quality of the extracted data before and after

[8] http://semanticweb.kmi.open.ac.uk
[9] http://kmi.open.ac.uk/news

Table 1. An experimental evaluation of the metadata extraction.

Type	People	Organizations	Projects	Total
Recall				
ESpotter	**0.815**	**0.783**	**0.761**	**0.798**
ASDI	**0.771**	**0.783**	**0.761**	**0.775**
Precision				
ESpotter	**0.873**	**0.667**	**1**	**0.787**
ASDI	**0.860**	**0.946**	**1**	**0.911**
F-measure				
ESpotter	**0.843**	**0.72**	**0.864**	**0.792**
ASDI	**0.813**	**0.856**	**0.864**	**0.837**

the verification process. Table 1 shows the evaluation results. Although the recall rate is slightly lower than Espotter (this is because PANKOW sometimes gives back empty classifications thus resulting in losing some correct values), the precision rate has been improved. The f-measure values show that ASDI performs better than ESpotter in terms of the overall performance. This means that the quality of the extracted data is improved by our verification engine.

7.2 Question answering

To assess the performance of the question answering facility, we carried out an initial study in the context of the KMi semantic web portal. We collected 69 different questions. Among them, 40 have been handled correctly. 19 more can be handled correctly if re-formatted by end user. This was a pretty good result, considering that no linguistic restrictions were imposed on the questions (please note that we have asked users not to ask questions which required temporal reasoning, as the underlying ontology does not cover it).

Among the failures, linguistic handling accounts for 16 out of the 20 failures, (23.18% of the queries). This is because the natural language processing component is unable to classify the query and generate appropriate intermediate representations (e.g., the question "what are the KMi publications in the area of the semantic web"). Such questions however can usually be easily reformulated by the end users and thus to be answered correctly. For example, changing the question mentioned earlier as "what are the KMi publications in the semantic web area", avoiding the use of nominal compounds (e.g, terms that are a combination of two classes as "akt researchers"). Another important observation is that some failures (10 out of 69) are caused by the lack of appropriate services over the ontology, e.g. the queries about "top researchers". Finally, the limitation of the coverage of the available metadata also plays an important role in the failures. For example, when dealing with the question "who funds the Magpie project", as there is no such information available in the metadata repository, the answer engine fails.

7.3 Semantic search

Although the semantic search engine is still in its infancy as the ranking algorithm has not yet been fully investigated, it produces encouraging results. Figure 6 shows the results of the search example which has been described in section 6. Behind the scene, the news stories are annotated by the metadata extraction component ASDI and thus being associated with semantic mark-ups. The metadata of phd students are extracted from the departmental databases.

As shown in the figure, the news entry which mentions most phd students appears in the top, as it gets the biggest relation weight. The news entries that mention other semantic entities (e.g., the news story ranked as the second mentions the project *AKT*) with which many phd students have relations also get good rankings. This is because in the experiment the ranking algorithm gives implicit relations half the weight of the explicit ones. This needs further investigation. The semantic search mechanism is however profound, which makes use of the available semantic relations between different resources to bring forward most relevant information.

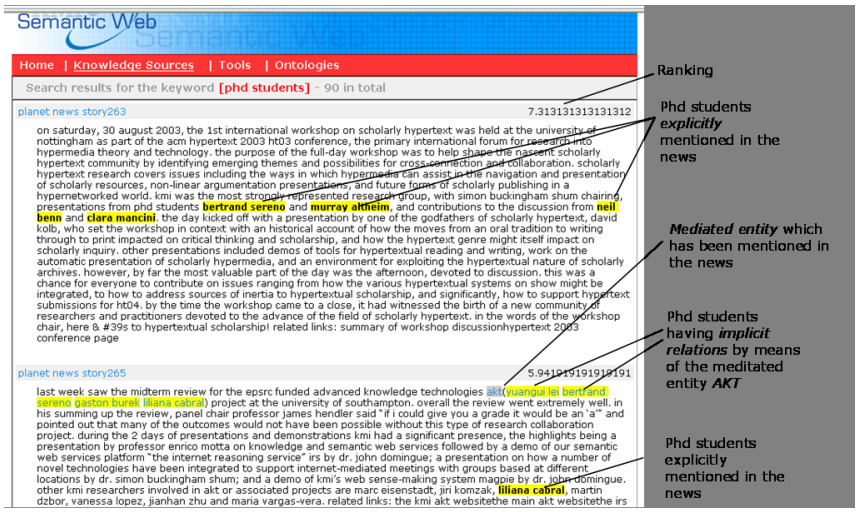

Fig. 6. A screenshot of the semantic search results of the example *"searching news about phd students"*.

8 Discussions

The core observation that underlies this paper is that, in the case of semantic web portals that offer both end users and applications an integrated access to information in specific user communities, it is crucial to ensure that i) high

quality metadata are acquired and combined from several data sources, and ii) simple but effective querying facilities are provided thus enabling knowledge to be accessed as easily as possible. By quality here we mean that the semantic data contains no duplicates, no errors and that the semantic descriptions correctly reflect the nature of the described entities. By simple but effective querying we mean that users are able to pose questions in their own terms and get precise results.

Our survey of a set of semantic web portals shows that on the one hand little or no attention is paid to ensure the quality of the extracted data, and on the other hand the support for data querying is limited. In contrast with these efforts, our semantic web portal infrastructure focuses on ensuring the quality of the extracted metadata and the facilities for data querying.

Our evaluation of the quality verification module shows that it improved the performance of the bare extraction layer. The metadata extraction component ASDI outperforms ESpotter by achieving 91% precision and 77% recall. In the context of semantic web portals, precision is more important than recall - erroneous results annoy user more than missing information. We plan to improve the recall rate by introducing additional information extraction engines to work in parallel with ESpotter. Such a redundancy is expected to substantially improve recall.

Our initial study of the question answering tool AquaLog shows that it provides reasonably good performance when allowing users to ask questions in their own terms. We are currently working on the failures learned from the study. We plan to use more than one ontology so that limitations in one ontology can be overcome by other ones.

The semantic search facility developed within the framework of our infrastructure takes advantage of semantic representation of information to facilitate keyword searching. It produces encouraging results. We plan to further investigate i) a more fine grained ranking algorithm which gives appropriate consideration for all the factors affecting the search results, and ii) the effect of implicit relations on search results.

We are, however, aware of *a number of limitations* associated with this semantic web portal infrastructure. For example, the manual specification of mappings in the process of setting up the metadata extraction component makes the approach heavy to launch. We are currently addressing this issue by investigating the use of automatic or semi-automatic mapping algorithms. A semi-automatic mapping would allow our tool to be portable across several different application domains.

Another limitation of the infrastructure is the support for *trust* management. As anyone can contribute to semantic data, trust is an important issue, which needs to be addressed. We plan to study this issue in the near future, by looking at i) how trust factors can be associated with the metadata of semantic web portals, ii) how they can be combined together when one piece of markup comes from different sources, and iii) what roles the processing tools play from the trust point of view.

Acknowledgements

We wish to thank Dr. Marta Sabou and Dr. Victoria Uren for their valuable comments on earlier drafts of this paper. This work was funded by the Advanced Knowledge Technologies Interdisciplinary Research Collaboration (IRC), the Knowledge Sharing and Reuse across Media (X-Media) project, and the OpenKnowledge project. AKT is sponsored by the UK Engineering and Physical Sciences Research Council under grant number GR/N15764/01. X-Media and OpenKnowledge are sponsored by the European Commission as part of the Information Society Technologies (IST) programme under EC Grant IST-FP6-26978 and IST-FP6-027253.

References

1. T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284(5):34 – 43, May 2001.
2. P. Cimiano, S. Handschuh, and S. Staab. Towards the Self-Annotating Web. In S. Feldman, M. Uretsky, M. Najork, and C. Wills, editors, *Proceedings of the 13th International World Wide Web Conference*, pages 462 – 471, 2004.
3. H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. Gate: A framework and graphical development environment for robust nlp tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*, Philadelphia, 2002.
4. C. Fellbaum. *WORDNET: An Electronic Lexical Database*. MIT Press, 1998.
5. R. Guha, R. McCool, and E. Miller. Semantic search. In *Proceedings of WWW2003*, 2003.
6. J. Heflin and J. Hendler. Searching the web with shoe. In *Proceedings of the AAAI Workshop on AI for Web Search*, pages 35 – 40. AAAI Press, 2000.
7. E. Hyvonen, E. Makela, M. Salminen, A. Valo, K. Viljanen, S. Saarela, M. Junnila, and S. Kettula. MuseumFinland – Finnish Museums on the Semantic Web. *Journal of Web Semantics*, 3(2), 2005.
8. Y. Lei. An Instance Mapping Ontology for the Semantic Web. In *Proceedings of the Third International Conference on Knowledge Capture*, Banff, Canada, 2005.
9. Y. Lei, E. Motta, and J. Domingue. Ontoweaver-s: Supporting the design of knowledge portals. In *Proceedings of the 14th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2004)*. Springer, October 2004.
10. V. Lopez, M. Pasin, and E. Motta. AquaLog: An Ontology-portable Question Answering System for the Semantic Web. In *Proceedings of ESWC*, 2005.
11. P. Mika. Flink: Semantic Web Technology for the Extraction and Analysis of Social Networks. *Journal of Web Semantics*, 3(2), 2005.
12. M.C. Schraefel, N.R. Shadbolt, N. Gibbins, H. Glaser, and S. Harris. CS AKTive Space: Representing Computer Science in the Semantic Web. In *Proceedings of the 13th International World Wide Web Conference*, 2004.
13. J. Zhu, V. Uren, and E. Motta. ESpotter: Adaptive Named Entity Recognition for Web Browsing. In *Proceedings of the Professional Knowledge Management Conference*, 2004.