Journal of Web Engineering, Vol. 6, No. 4 (2007) 283–308
© Rinton Press

AN INFRASTRUCTURE FOR SEMANTIC WEB PORTALS

YUANGUI LEI, VANESSA LOPEZ, ENRICO MOTTA, VICTORIA UREN
Knowledge Media Institute, the Open University, Walton Hall
Milton Keynes, MK7 6AA, United Kingdom
{*y.lei, v.lopez, e.motta, v.s.uren*}*@open.ac.uk*

Received June 1, 2007
Revised December 1, 2007

This paper presents our Semantic Web portal infrastructure, which focuses on how to enhance knowledge access in traditional Web portals by gathering and exploiting semantic metadata. Special attention is paid to three important issues that affect the performance of knowledge access: i) *high quality metadata acquisition*, which concerns how to ensure high quality while gathering semantic metadata from heterogeneous data sources; ii) *semantic search*, which addresses how to meet the information querying needs of ordinary end users who are not necessarily familiar with the problem domain or the supported query language; and iii) *semantic browsing*, which concerns how to help users understand and explore the problem domain.

Keywords: Semantic Web Portal, Semantic Metadata Acquisition, Quality Control, Semantic Search, Semantic Browsing

1 Introduction

Knowledge access in traditional Web portals is typically achieved by i) hyperlinks in Web resources, which enable navigation across Web resources; and ii) text search, which allows the user to search information contained in the underlying data sources. The performance is greatly limited by the lack of explicit meaning of data sources. For example, when searching for news stories about *PhD students*, with traditional Web portals, we often can only get news entries in which the term "PhD students" appears. Those entries which mention the names of students but do not use the term directly will be missed out. Such news entries however are often the ones that the user is interested in.

The goal of Semantic Web (SW) portals is to address this issue by adding and exploiting semantic metadata that describe the meaning of data sources. One crucial condition for achieving this goal is the existence of appropriate facilities that would ensure *high quality* metadata to be gathered from the underlying data sources, thus enabling the associated services, in particular, knowledge access services, to produce accurate results. Several problems can happen in the metadata extraction process, which decrease the quality of the semantic metadata. Data sources can have low quality (e.g., misspelling, erroneous statements, etc.); problems can be derived from the fusion of multiple data sources (e.g., inconsistencies and duplicated entries) or be introduced by the tools employed. Changes to the data sources can also bring problems. Therefore, measures for quality control are essential.

Another important condition for successful SW portals is the existence of facilities that provide sufficient support to help end users achieve their information seeking goals. Essentially, this requires two types of support. One is to help users make explicit queries and exploit semantic metadata to produce precise answers. We call this *semantic search*. The other is to supply end users with browsing facilities to enable them explore the problem domain and find the information they are looking for, which we call *semantic browsing*.

Regarding semantic search, an important issue is how to approach query specification for ordinary end users in a way that would lower the barrier for them to make complex semantic queries. In the case of relatively simple queries, a straightforward way of asking queries is highly desirable. Both natural language question answering and keyword based searching are elegant solutions to the problem. In the situation where complex queries are involved, intuitive user interfaces are needed to support the specification.

As to semantic browsing, a key issue is how to support both coarse grained and fine grained views of the problem domain. While coarse grained views present general pictures of the entire knowledge space of a SW portal, fine grained views display more detailed pictures of the given individual entity. Both types of views are beneficial for end users in understanding the problem domain.

Our overview of existing SW portals with respect to these issues (in Section 2) reveals that there is plenty of space for improvement. In this context, we proposed and implemented a SW portal infrastructure, called KSW, which offers several tools to enhance the performance of SW portals on the issues clarified above.

The rest of the paper is organized as follows. We begin by investigating how existing SW portals approach the issues of metadata extraction and knowledge access (Section 2). We then present an overview of our infrastructure and the KMi context that is needed to understand the examples of the paper (Section 3). Thereafter, we explain the core components of the infrastructure in sections 4, 5, 6, and 7. In Section 8, we describe the related work. Finally, we conclude our paper with a discussion of our main contributions, limitations of the approach and future work in Section 9.

2 SW Portals: State of the Art

In this section, we investigate how existing *SW portals* approach the three important issues described above, namely high quality metadata extraction, semantic search, and semantic browsing. As our special attention of this section is paid to SW portals, general related work is described later in Section 8. We survey a representative sample of portals and tools without performing an exhaustive study of this research strand.

2.1 An overview

MindSwap[a], Esperonto[b], OntoWeb[c], and Knowledge Web[d] are examples of the first generation of SW portals that are designed for sharing data sources among partners of specific research projects. Metadata in these portals are typically constructed manually. As such, quality control is usually done manually. Semantic search is enabled by form based ontology search, which uses the structures of the underlying ontology to generate forms, menus and drop

[a]http://www.mindswap.org/
[b]http://www.esperonto.net/
[c]http://www.ontoweb.org/
[d]http://knowledgeweb.semanticweb.org

down lists, thus guiding the user in formulating queries. Some portals (e.g., MindSwap) also support keyword based semantic search, which returns semantic entities that are related to the keyword. As to semantic browsing, tree view based visualization techniques are common in these portals, which project the collections of the gathered metadata and data sources into trees using the domain ontology. Tools for building such Web portals (e.g., SEAL [1], OntoWebber [2] and ODESeW [3]) typically focus on how to automatically generate Web pages that support the main functionalities, namely metadata construction, form based search, and content visualization.

Compared to the SW portals described above, CS AKTive Space [4], Flink [5], MuseumFinland [6], mSpace [7], and the MultiMediaN E-Culture Demonstrator [8] are more recent examples that are equipped with more sophisticated mechanisms to demonstrate the advantages of Semantic Web technologies in specific problem domains.

CS AKTive Space gathers semantic metadata automatically on a continuous basis for the UK's Computer Science research domain. Quality control related issues such as the problem of duplicate entities are addressed by heuristics based methods or by using manual input. Semantic browsing is supported by a geographic visualizer, which provides an interactive geographic browsing facility. As to semantic search, the portal offers end users an RDF query language based interface, which requires users to specify queries using the supported query language (e.g., SPARQL[e]).

Flink focuses on online social networks in the research community of the Semantic Web. Metadata are gathered from Web pages, FOAF profiles, Emails, and Web databases (e.g., Google Scholar). Quality control is addressed by two means. One is co-relation, which relies on a set of domain specific inference rules to determine whether different resources refer to the same individual. The other is merging, which combines profile information based upon the *owl:sameAs* relation. Regarding semantic browsing, graphs are used to support the visualization and the navigation of social networks of individuals. Semantic search is however not addressed.

MuseumFinland offers integrated access to heterogeneous museum collections. Semantic metadata are gathered by means of mapping database schemas to the shared museum ontologies. Errors were logged by the system for correction by a human user. Semantic browsing is supported by a view-based multi-facet browsing facility, which makes use of museum category hierarchies to filter information. Semantic search, on the other hand, is addressed by a keyword search facility, which matches the keyword with the available categories and then uses the category matches to filter information.

mSpace is a SW portal specialized in classical music. It offers users a form-like user interface, which allows users to organize the information space according to their querying requirements by selecting or deselecting columns that represent concepts of the domain ontology.

MultimediaN E-culture Demonstrator pays special attention to the indexing and the search of large collections of resources contained in the Dutch cultural-heritage domain. Metadata are gathered mainly automatically by i) transforming the existing textual annotations into annotations using the domain ontology and ii) clustering resources into the pre-defined semantic categories. Manual annotation tools are also provided to support users annotating

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

any resources on the Web. A keyword based semantic search facility is provided, which returns art works that are related to the keyword. These art works are clustered using the relevant semantic terms. Regarding semantic browsing, a traditional view-based approach is employed.

2.2 Summary

High quality metadata acquisition. Most recent SW portals mentioned above support metadata acquisition. However, 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 and is often domain specific. For example, how to detect and address spurious annotations has not been tackled in any of the approaches mentioned above. Such problems may significantly decrease the quality of the acquired semantic data.

Semantic search. This has been addressed in most SW portals in one way or another. Several search modes are observed, including keyword based, form based, view based, and RDF query language based search. While each user interaction mode has its unique advantages over the others, the user support provided is not sufficient when each is used alone for dealing with complex search environments. For example, keyword search and natural language querying offer easy routes in for users, but don't support exploration of the semantic search space. Those users who are not necessarily familiar with the problem domain are left out of the systems. View based search and forms can help the user explore the space, but become tedious to use in large spaces and impossible in heterogeneous ones.

Semantic browsing. The most common support found is tree based browsing, which projects the underlying data (or data sources) into hierarchies using the domain ontology. Some tools offer more comprehensive support by generating i) facet based views (e.g., MuseumFinland, mSpace, and MultimediaN), which cluster data sources into different facets; ii) geographic views (e.g., in CS AKTive Space), which support information browsing according to the specified geographic locations; or iii) graph views (e.g., Flink), which allow users to navigate through the knowledge space by working with graphs. Apart from the support for such coarse grained views of the problem domain, most SW portals also offer means to allow the viewing of the detailed specification of a given individual entity.

While these means are quite useful in helping users understand the problem domain, fine grained level support is, however, not sufficient. In most cases, users are more likely to be interested in the semantic relation network of the given entity with its neighborhood entities than in the detailed specification. Simply displaying the details of the given entity does not help much, as many relations are not directly defined in the specification. For instance, we cannot get relations between Enrico and his colleagues by viewing the details of the instance *Enrico*. As a consequence, users have to derive the knowledge networks themselves in order to get an appropriate view. A partial exception is Flink, which focuses on visualizing the knowledge networks between people. Even with Flink, the user can only browse generic connections between different people without any support for fine grained classification (i.e., the distinguishing of different types of relations, e.g., knowing, co-authoring).

3 An Overview of the KMi SW Portal Infrastructure

Our SW portal infrastructure addresses the important limitations of existing SW portals identified above by offering several tools which enable comprehensive support for knowledge access. First, an automated metadata acquisition tool, *ASDI* [9], is developed which focuses on how to ensure high quality in the metadata acquisition process. Second, two semantic search tools are implemented, which aim to hide the complexity of semantic search from end users and support users making complex queries. One is *AquaLog* [10], which answers questions submitted in natural language format. The other is *SemSearch* [11], which offers several ways to help end users make complex semantic queries. Third, the infrastructure supports a semantic browser called *SemBrowser*, which facilitates the browsing of the semantic networks contained in the underlying data sources.

Figure 1 shows an overview of the KSW architecture. Essentially, KSW produces SW portals that are equipped with the functionalities described above. It adopts a layered architecture, containing a source data layer, an extraction layer, a semantic data layer, a semantic service layer, and a presentation layer.

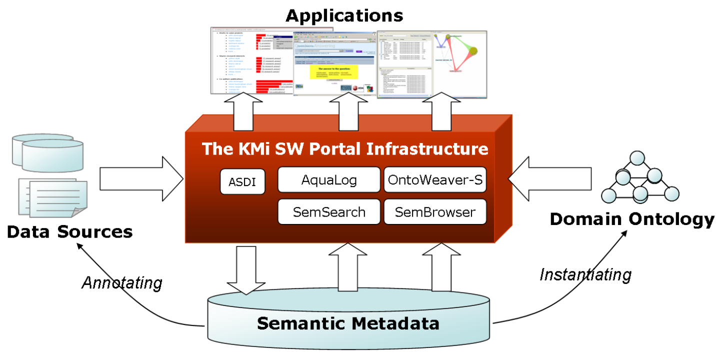


Fig. 1. Building Semantic Web portals with the KMi SW portal infrastructure.

- **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** offers knowledge access services over the semantic metadata repositories for SW portals. Central to this layer are three components: the semantic search tools AquaLog and SemSearch, and the semantic browsing tool SemBrowser. As will be explained in Section 5 and Section 6, the semantic search tools address the limitations associated with the semantic search facility of existing SW portals. SemBrowser, on the other hand, employs several means to enable the browsing of knowledge networks of individual entities. SemBrowser will be described in Section 7.

- **The presentation layer** supports the generation of user interfaces for SW portals. Our SW portal infrastructure relies on OntoWeaver-S [12] 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. Please refer to [12] for the details.

We have designed and tested KSW in the context of building a SW portal for KMi that would provide integrated access to various aspects of the academic life of our lab[f]. The relevant knowledge is distributed in several different data sources such as departmental databases (e.g., information about people, technologies, projects, etc.) and HTML pages (e.g., news stories). In particular, KMi has an electronic newsletter[g], which now contains an archive of several hundred news items, describing events of significance to the KMi members. Beside its heterogeneous nature, another important feature of the KMi domain data is that it continuously undergoes changes. For example, KMi-related events are reported in the newsletter and added to the news archive. Therefore, the semantic metadata that underly the portal have to be updated often. The KMi SW portal has been up and running for a couple of years now, gathering and maintaining semantic metadata from data sources and offering knowledge access facilities to both human users and SW applications.

4 ASDI: Gathering High Quality Metadata from Heterogeneous Sources

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 the 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 assurance related tasks. Figure 2 shows its architecture. ASDI relies on an *automatic 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.

[f]http://semanticweb.kmi.open.ac.uk
[g]http://kmi.open.ac.uk/news

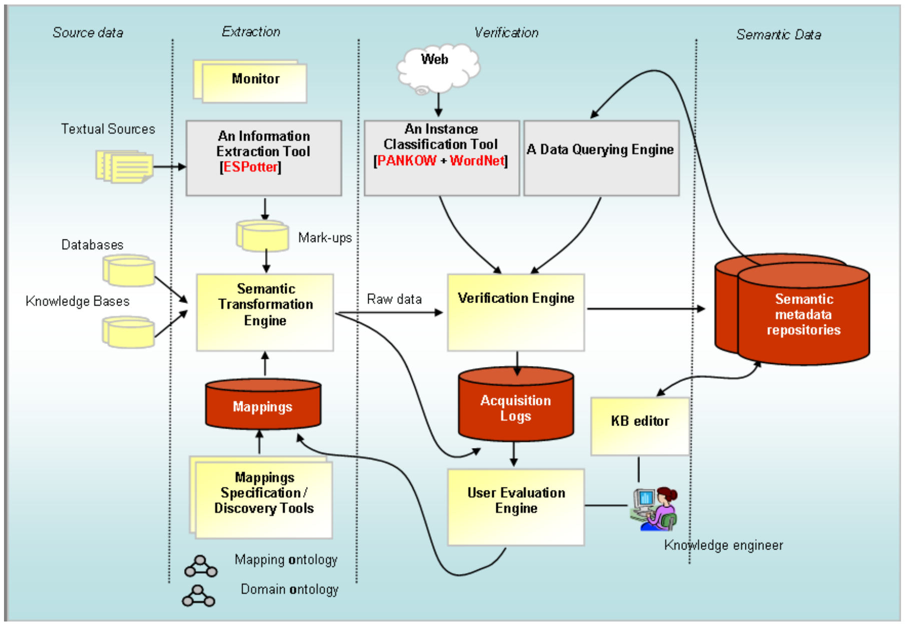


Fig. 2. The architecture of the metadata extraction tool ASDI.

4.1 Information extraction

We use ESpotter [13], a named entity recognition (NER) system, to provide an information extraction service. ESpotter accepts the URL of a textual document as input and produces a list of the named entities mentioned in that text. In particular, ESpotter is able to provide an adaptive service by utilizing ontologies and lexicon resources that are available in the problem domain. For example, in 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 [14]) 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 has been 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.

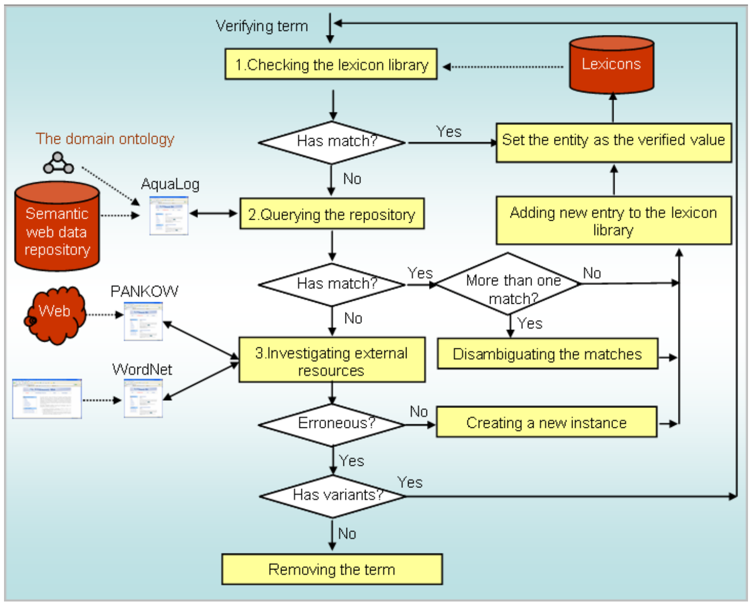

Verifying term
1.Checking the lexicon library
Lexicons
Has match?
Yes
No
Set the entity as the verified value
The domain ontology
AquaLog
Semantic web data repository
2.Querying the repository
Adding new entry to the lexicon library
Has match?
Yes
More than one match?
No
Yes
No
PANKOW
Web
3.Investigating external resources
Disambiguating the matches
WordNet
Erroneous?
No
Creating a new instance
Yes
Has variants?
Yes
No
Removing the term

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

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.

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 metadata repository. This step uses an ontology-based data querying engine, to query the already acquired semantic metadata (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 de-reference 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's name is however misspelled as it should be *Dnyanesh Rajpathak*. While the 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 falls down 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 ambiguity. The verification engine exploits the semantic relations between the entities that appear in the same piece of text (e.g. the news story) and the matches as 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 extracted from the same document and checks whether they have any relation by using formal queries. In this example, the instance *AKT* is annotated in the same story. Using a Sesame Serql query[h] "*select r from* {*ksw:Victoria-Uren*} *r* {*ksw:Victoria-Uren*}", we can get a relation *has-project-member*) between the *Victoria-Uren* and *AKT*. *Victoria Wilson*, on the other hand, does not have any direct relation with *AKT*. 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, in which case it should be removed, or correct but new to the system. For this purpose, an instance classification tool was developed, which makes use of PANKOW [15] and WordNet [16], 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 by the information extraction tool (i.e., "organization" in this example) 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.

In summary, ASDI provides several means to address the issue of high quality metadata extraction clarified in the introduction. First, it addresses ambiguities by taking into account the context in which an entity is mentioned in order to determine its type. Second, it contains

[h]http://www.openrdf.org/doc/sesame/users/ch06.html

a verification engine that checks the validity of any derived metadata against a repository of trusted domain knowledge and against the information available on the Web. Finally, since the whole acquisition process is automatic, it can be automatically run whenever new data becomes available, thus ensuring that the semantic metadata is always up to date.

5 AquaLog: Enabling Queries in Natural Language

As mentioned in the introduction, natural language (NL) question answering is an elegant approach, which provides users with a straightforward way of posing queries, without having to learn the vocabulary adopted by the ontology or having to master a special query language. AquaLog exploits this approach to facilitate information querying for SW portals.

To bridge the gap between the user terminology and the underlying domain ontology of SW portals, AquaLog exploits a variety of linguistic mechanisms, including GATE [17], distance metrics [18], lexical resources (WordNet [16]), and the structure of the domain ontology. Furthermore, to ensure that the system adapts to the jargons of users, thus leading to improved performance over time, AquaLog is coupled with a sophisticated learning mechanism.

Figure 4 shows an overview of the AquaLog architecture. The core components are i) *a linguistic component*, which parses the natural language queries into a set of linguistic triples; ii) *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 iii) *an answers engine*, which infers answers from the derived ontological triples.

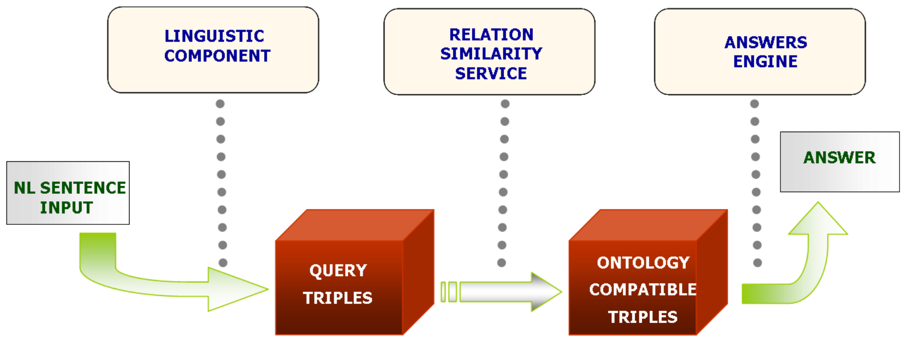


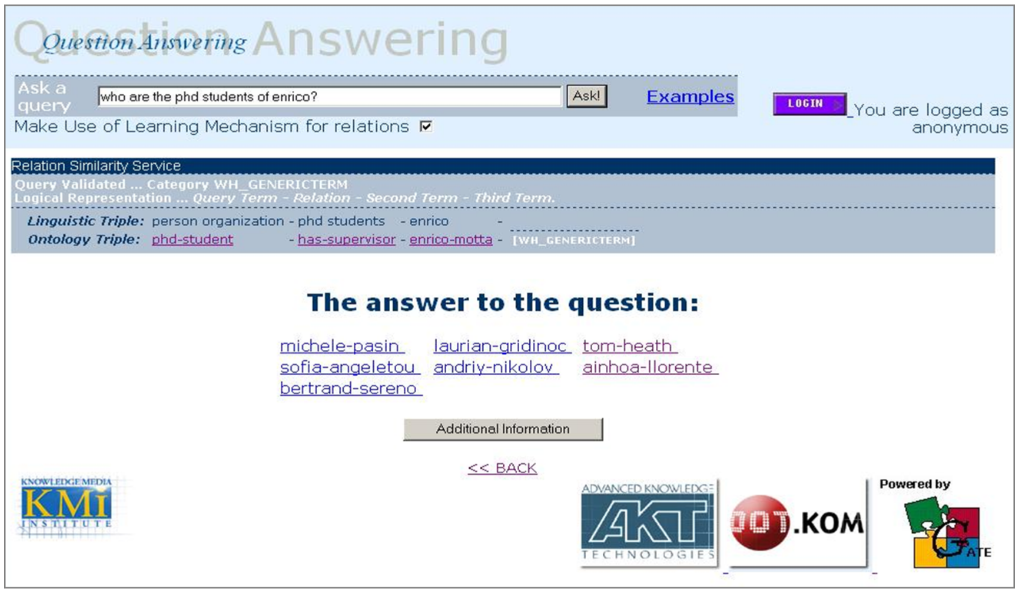
Fig. 4. An overview of the AquaLog architecture.

AquaLog algorithms are detailed in [10]. Here we illustrate its usage as an integrated component of the presented infrastructure by walking through two concrete examples. Consider the questions “who are the PhD students of Enrico” and “who are the PhD students of Enrico that are doing research on trust”. In order to be competent to understand the questions and give precise answers, behind the scenes, the question answering mechanism in AquaLog takes the following steps:

Step 1: Parsing a NL query into linguistic querying triples. The linguistic component maps the NL input query into a set of linguistic triples. The notation used to represent a triple is *(term, relation, term)*. At this stage the analysis is domain independent and purely based on linguistic criteria. For instance, the first user query described above is parsed into *(person/organization, PhD students, enrico)*, while the second one is parsed into *(person/organization, PhD students, enrico)* and *(which is, doing research, trust)*.

Step 2: Interpreting linguistic querying triples as ontology-compliant triples. First, the RSS component maps user query terms to entities contained in the target ontology by making use of i) string metric algorithms, ii) lexical resources such as WordNet, or lexicon libraries, if any. Once one or more query terms are successfully mapped into possible candidate ontology terms, the RSS component then uses the ontology to relate them together and to produce the ontology compliant triples that better represent the user query.

For the first linguistic triple*(person/organization, PhD students, enrico)*, the RSS component matches the term *Enrico* to the instance *Enrico-Motta* which is of the class *kmi-staff-member*; and the term *PhD students* to the class *phd-students*. RSS re-classifies the triple and now the problem becomes finding the relation that links the class *phd-student* to the class *kmi-staff-member* or vice versa. RSS achieves this task by using the subsumption hierarchy, in which all the super classes and subclasses of both terms are considered as domain and range of the potential relations. In this example, there is only one ontological relation *has-supervisor* that makes sense between the terms. Therefore the RSS component generates the ontological triple *(phd-student, has-supervisor, enrico-motta)*. In this particular case, the domain information enclosed in the ontology is enough to interpret the query without requiring user feedback. The results can be seen in Figure 5.



who are the phd students of enrico?

Make Use of Learning Mechanism for relations ☑

Relation Similarity Service
Query Validated ... Category WH_GENERICTERM
Logical Representation ... Query Term - Relation - Second Term - Third Term.
Linguistic Triple: person organization - phd students - enrico -
Ontology Triple: phd-student - has-supervisor - enrico-motta - [WH_GENERICTERM]

The answer to the question:

michele-pasin laurian-gridinoc tom-heath
sofia-angeletou andriy-nikolov ainhoa-llorente
bertrand-sereno

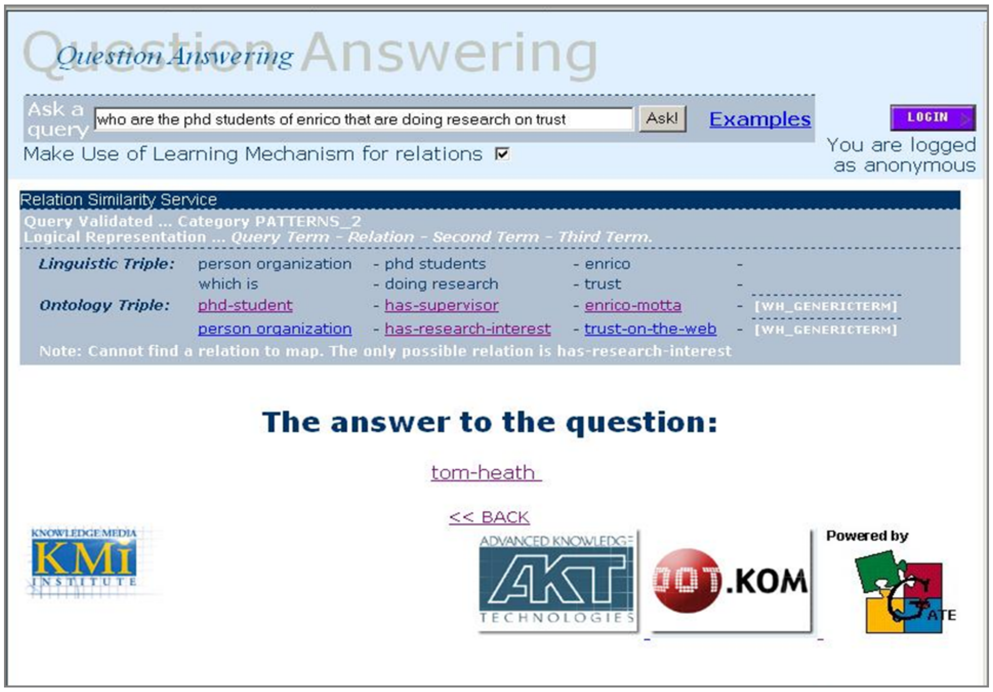
Additional Information
<< BACK
KNOWLEDGE MEDIA KMi INSTITUTE
ADVANCED KNOWLEDGE AKT TECHNOLOGIES
DOT.KOM
Powered by GATE

Fig. 5. A basic query example in AquaLog.

Likewise, the second linguistic triple *(which is, doing research, trust)* is linked to the first triple through the non-ground term *phd-student* and processes as the ontology triple *(phd-student, has-research-interest, trust)*. The resultant ontological triples and results can be seen in Figure 6. Then, the user can click on all the ontological terms to navigate through the ontology.

Since the universe of discourse is determined by the particular ontology, in the case that there are discrepancies between the NL questions and a set of possible relations recognized in the ontology, external resources like WordNet are used to handle unknown vocabulary (terms) by giving a set of synonyms. When neither the ontology, not the lexical resources are enough to disambiguate between possible ontology relations, the user is asked to disambiguate. The learning mechanism will then update the lexical resources to ensure that for a given ontology and a specific user community, its performance improves over the time, as the users' feedback allows AquaLog to learn novel associations between the relations used by users (in the context of a query triple) and the internal structure of the ontology.

Step 3: Inferring answers from ontology-compliant triples. The answers engine retrieves the available metadata that satisfy the derived ontology triples to produce answers depending on the query category. As shown in Figure 6, the answer to our last query example is an instance of the class phd-stduents, which has a research interest in trust and at the same time has Enrico Motta as supervisor from the context of the KMi SW portal.



who are the phd students of enrico that are doing research on trust

Make Use of Learning Mechanism for relations ✓

You are logged as anonymous

Relation Similarity Service

Query Validated ... Category PATTERNS_2
Logical Representation ... *Query Term - Relation - Second Term - Third Term.*

Linguistic Triple:	person organization	- phd students	- enrico	-
	which is	- doing research	- trust	-
Ontology Triple:	phd-student	- has-supervisor	- enrico-motta	- [WH_GENERICTERM]
	person organization	- has-research-interest	- trust-on-the-web	- [WH_GENERICTERM]

Note: Cannot find a relation to map. The only possible relation is has-research-interest

The answer to the question:

tom-heath

<< BACK

Fig. 6. A complex query example in AquaLog.

A distinctive feature of Aqualog is that it is ontology independent. Its performance strongly depends on the quality of the ontology and the gathered metadata. This again emphasizes the importance of the quality of semantic metadata. End user support is enhanced in the latest development by allowing the user i) to select an ontology from a pre-defined subset of ontologies or ii) to upload their own ontologies to answer queries.

6 SemSearch: Helping Users Make Complex Queries

The role of SemSearch in our SW portal infrastructure is to address the two important issues associated with semantic search namely query specification and relation search. Figure 7 shows an overview of the SemSearch architecture. As shown in the figure, SemSearch uses keyword based semantic search as a starting point for complex queries. Although it is iterative, the search process of SemSearch can be classified into five major steps:

- **Step 1.** Interpreting the user query, which is to find out the semantic entity matches for the keywords involved in a user query.
- **Step 2.** Translating the user query into formal queries.
- **Step 3.** Refining queries. This step is optional, as refinement may not be necessary in some queries.
- **Step 4.** Querying the back-end semantic data repositories using the generated formal queries.
- **Step 5.** Ranking the querying results.

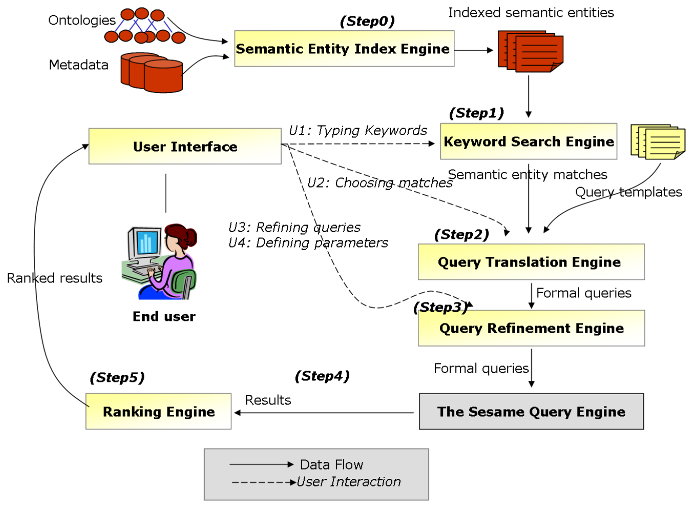


Fig. 7. An overview of the SemSearch architecture.

The core components of SemSearch are: i) *a query interface*, which supports the specification of multi-keywords queries; ii) *a keyword search engine*, which makes sense of user queries by exploiting the domain ontology and the extracted metadata; iii) *a query translation engine*, which translates user queries into formal queries; iv) *a query refinement engine*, which refines user queries according to the specified requirements; v) *a ranking engine*, which presents the search results in an order that indicates the degree to which they satisfy the user query; and vi) *a user interface component*, which supports all the user interaction required for query specification and refinement, and results presentation.

The system also provides an index engine, which indexes semantic entities contained in the domain ontology and the gathered metadata repositories using Lucene[i]. The indexing step takes place before semantic search for the sake of speeding up the search process (which is why we refer it as step 0). SemSearch relies on the Sesame query engine[j] to perform queries against the metadata repositories of SW portals.

As our main focus here is on query specification, we concentrate on those components that are highly related, including the SemSearch query interface, the keyword search engine, the query translation engine, and the user interface component. Other components are out of the scope of the paper.

To illustrate the SemSearch system, we use two query examples which apply to the KMi context. One is querying for news stories that are relevant to PhD students. The other example is looking for relations between *John* and *Enrico*.

The SemSearch Query Interface. The query interface extends traditional keyword search languages by allowing the explicit specification of i) the queried subject which indicates the type of the expected search results, and ii) the combination of keywords. The query interface uses the operator ":" to capture the query subject and the operators "and" and "or" to specify the combination of keywords (apart from the subject keyword). A user query in SemSearch looks like "*subject:keyword1 and/or keyword2 and/or keyword3 ...*".

With this syntax, the first query example can be specified as *(news: PhD students)*, where the term *news* is the query subject and the term *PhD students* is a required keyword. Equipped with support for multiple keyword specification, the query interface can be easily used to specify relation search, where there are two keywords involved and both keywords refer to individual entities (i.e., instances). The second query example described above falls into this category and can be specified as *"John:Enrico"*.

The Keyword Search Engine. This component looks up the indexed metadata repositories for matches. The main search source is the *labels* of semantic entities. The rational for this choice is that, from the user point of view, labels often catch the meaning of semantic entities in an understandable way. In the case of instances, we also used their short literal values as the search source. So that when the user is searching for "chief scientist", the instance that has such a string as a value of its properties can be reached.

The matches are ranked by measuring their closeness to the keyword. Three factors have been taken into account, including *similarity*, *domain context*, and *query context*. The similarity factor measures the closeness of a match to the keyword syntactically (i.e., string similarity), which is derived from the text search engine employed. In future, we will add mechanism to compute the semantic similarity of the match to the keyword.

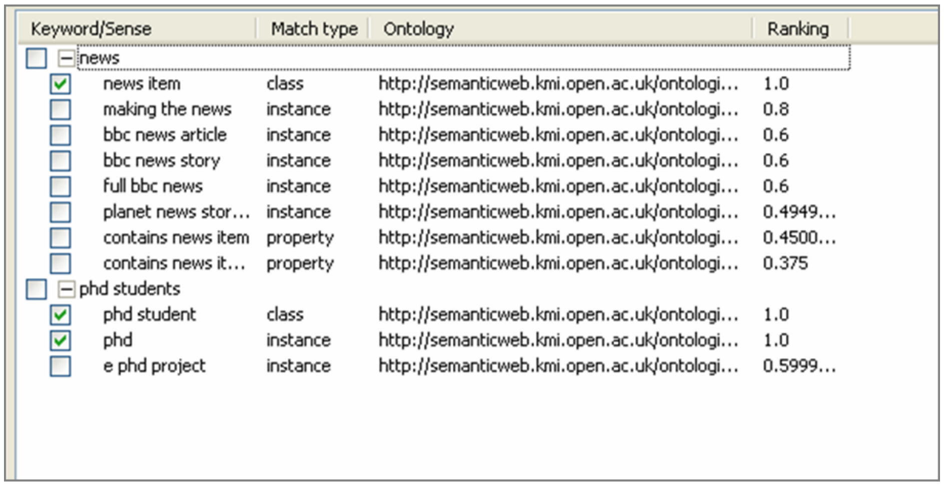
The domain context factor helps decide the closeness of the matches to the keyword from the specific domain point of view. For example, with the keyword "enric", is the user more likely to mean the person *enrico-motta* than the project *enrich*? It is calculated as $\frac{Pm_x}{\sum_{i=1}^{n} Pm_i}$, where Pm_x denotes the count of the match 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-exact matches.

The query context factor takes the position of a query term and its expected match type (i.e., class, instance, and property) into account when ranking the match results. Several

[i] http://lucene.apache.org/
[j] http://www.openrdf.org/

heuristic rules are used to rank the matches. For instance, the subject keyword (i.e., the first keyword in a user query) is assumed to refer more often to a class than an individual entity. Thus, for a subject keyword, class matches would get higher ranking.

For the sake of simplicity, we treat all three factors as equally important in ranking at the moment. Hence, the formula is read as r=$(r_s+r_d+r_q)/3$, where r_s denotes the match similarity; r_d is the domain context factor; and r_q depicts the query context factor. Nevertheless, it should be easy to assign weights to different factors in specific context. Figure 8 shows the matches of the keywords contained in the first example.



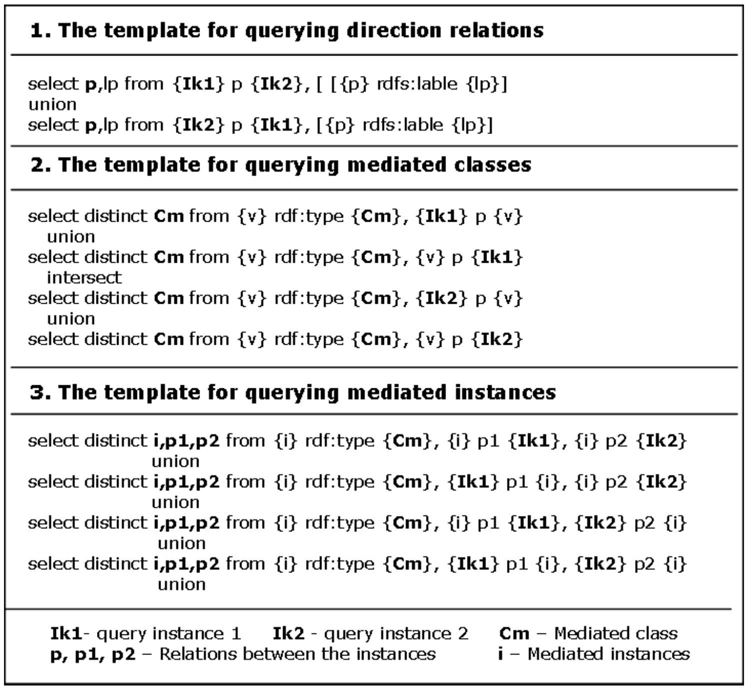
Keyword/Sense	Match type	Ontology	Ranking
☐ news			
✓ news item	class	http://semanticweb.kmi.open.ac.uk/ontologi...	1.0
☐ making the news	instance	http://semanticweb.kmi.open.ac.uk/ontologi...	0.8
☐ bbc news article	instance	http://semanticweb.kmi.open.ac.uk/ontologi...	0.6
☐ bbc news story	instance	http://semanticweb.kmi.open.ac.uk/ontologi...	0.6
☐ full bbc news	instance	http://semanticweb.kmi.open.ac.uk/ontologi...	0.6
☐ planet news stor...	instance	http://semanticweb.kmi.open.ac.uk/ontologi...	0.4949...
☐ contains news item	property	http://semanticweb.kmi.open.ac.uk/ontologi...	0.4500...
☐ contains news it...	property	http://semanticweb.kmi.open.ac.uk/ontologi...	0.375
☐ phd students			
✓ phd student	class	http://semanticweb.kmi.open.ac.uk/ontologi...	1.0
✓ phd	instance	http://semanticweb.kmi.open.ac.uk/ontologi...	1.0
☐ e phd project	instance	http://semanticweb.kmi.open.ac.uk/ontologi...	0.5999...

Fig. 8. The semantic entity matches of the keywords involved in "News:PhD Students".

The Query Translation Engine. This component translates user queries into formal (RDF) queries. It combines the matches of all the keywords and generates sensible queries according to the nature of the matches (e.g., classes, properties, or instances). A key operational problem is that in real world situations there can be a large number of matches and hence many more combinations. We used several heuristic rules to reduce the number of matches for each keyword. An algorithm has been developed to support query construction. Please refer to [11] for details.

Relation search, in which we search for either direct triples between two instances that match a pair of keywords or a pair of triples that link the instances via a mediating entity, is supported in the tool to some extent. Now let us demonstrate how it is acheived, using the query "John:Enrico" as an example. Two major steps are involved in the search process:

- *Step a. Retrieving direct relations between the two given instances.* Such relations are physically stored in the metadata repository. In the relation search example described above, such relations take the format *(Enrico, relation, John)* or *(John, relation, Enrico).*
- *Step b. Retrieving indirect relations.* The search engine first looks for mediating classes whose instances are linked to both instances. It then retrieves the mediating instances and their relations to the query terms.



1. The template for querying direction relations

```
select p,lp from {Ik1} p {Ik2}, [ [{p} rdfs:lable {lp}]
union
select p,lp from {Ik2} p {Ik1}, [{p} rdfs:lable {lp}]
```

2. The template for querying mediated classes

```
select distinct Cm from {v} rdf:type {Cm}, {Ik1} p {v}
   union
select distinct Cm from {v} rdf:type {Cm}, {v} p {Ik1}
   intersect
select distinct Cm from {v} rdf:type {Cm}, {Ik2} p {v}
   union
select distinct Cm from {v} rdf:type {Cm}, {v} p {Ik2}
```

3. The template for querying mediated instances

```
select distinct i,p1,p2 from {i} rdf:type {Cm}, {i} p1 {Ik1}, {i} p2 {Ik2}
                 union
select distinct i,p1,p2 from {i} rdf:type {Cm}, {Ik1} p1 {i}, {i} p2 {Ik2}
                 union
select distinct i,p1,p2 from {i} rdf:type {Cm}, {i} p1 {Ik1}, {Ik2} p2 {i}
                 union
select distinct i,p1,p2 from {i} rdf:type {Cm}, {Ik1} p1 {i}, {Ik2} p2 {i}
                 union
```

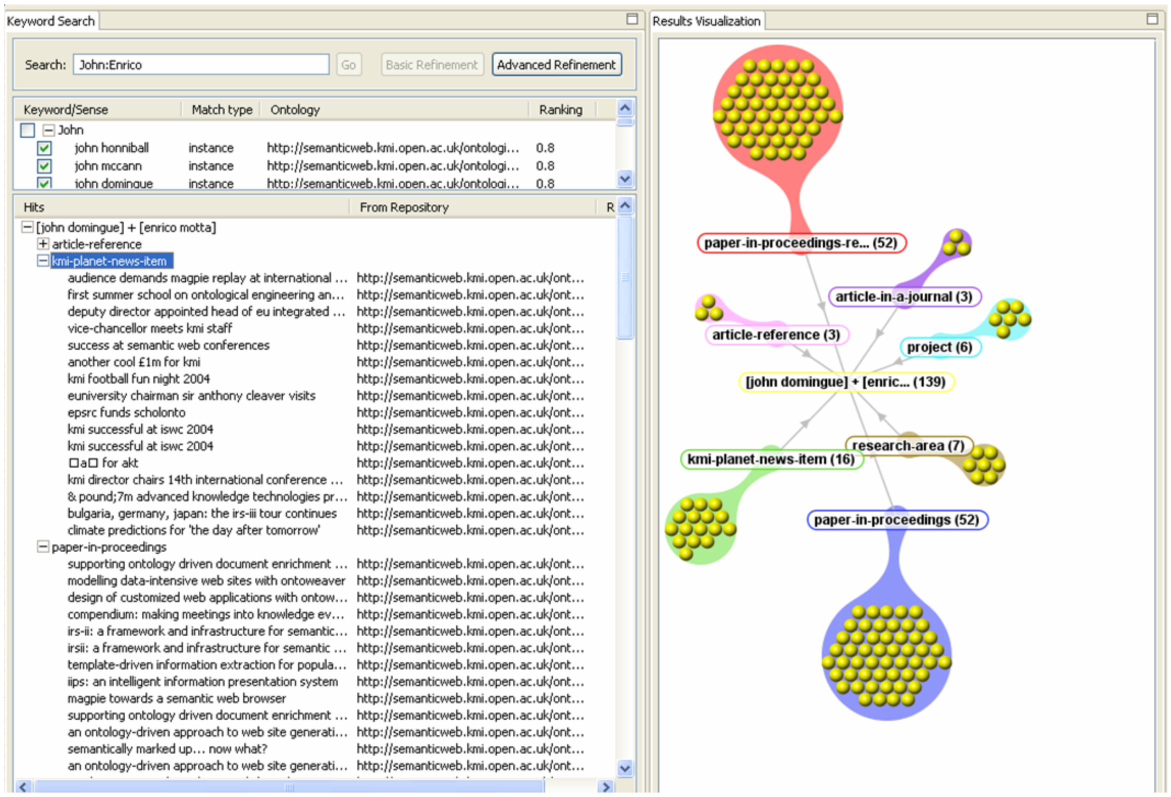
Ik1- query instance 1 **Ik2** - query instance 2 **Cm** – Mediated class
p, p1, p2 – Relations between the instances **i** – Mediated instances

Fig. 9. Templates for constructing queries for supporting relation search.

Spread activation techniques [19, 20] could be used here to walk through the graphs of the query instances and find the relations that are highly relevant. In the current version of the SemSearch prototype, we only consider relations that involve *one* mediating entity. The rationale behind this is that the strength of relations decays quickly as the length of their path grows. We treat all the mediating relations as equally important, as it is not possible to assign each relation a weight in real world scenarios.

Three templates are developed to support the generation of queries involved in relation search. Figure 9 shows a simplified version of the templates represented in the Sesame Serql query language. The first one supports the task involved in the first step. The other two templates work for the second step. Figure 10 shows the clustered results of the relation search example *"John:Enrico"*. The mediated classes include *Project*, *article-in-journal*, *kmi-planet-news-items*, *paper-in-proceedings*, etc. The search results (i.e., mediating instances and their relations) are grouped using the mediating classes.

The User Interface Component. This component supports all the interactions required for i) formulating and refining user queries and ii) presenting search results. A screen snapshot of the main user interface has already been shown in Figure 10. Four major facilities are provided by the main user interface, including i) allowing the specification of keyword queries; ii) displaying the matches of the keywords; iii) allowing the ticking on/off of keyword matches to refine user queries; and iv) displaying the final search results returned by the Sesame Query Engine. Two types of views are generated to help users understand the search



Search: John:Enrico

Keyword/Sense	Match type	Ontology	Ranking
☐ ⊟ John			
✓ john honniball	instance	http://semanticweb.kmi.open.ac.uk/ontologi...	0.8
✓ john mccann	instance	http://semanticweb.kmi.open.ac.uk/ontologi...	0.8
✓ john domingue	instance	http://semanticweb.kmi.open.ac.uk/ontologi...	0.8

Hits	From Repository	R
⊟ [john domingue] + [enrico motta]		
⊞ article-reference		
⊟ kmi-planet-news-item		
audience demands magpie replay at international ...	http://semanticweb.kmi.open.ac.uk/ont...	
first summer school on ontological engineering an...	http://semanticweb.kmi.open.ac.uk/ont...	
deputy director appointed head of eu integrated ...	http://semanticweb.kmi.open.ac.uk/ont...	
vice-chancellor meets kmi staff	http://semanticweb.kmi.open.ac.uk/ont...	
success at semantic web conferences	http://semanticweb.kmi.open.ac.uk/ont...	
another cool £1m for kmi	http://semanticweb.kmi.open.ac.uk/ont...	
kmi football fun night 2004	http://semanticweb.kmi.open.ac.uk/ont...	
euniversity chairman sir anthony cleaver visits	http://semanticweb.kmi.open.ac.uk/ont...	
epsrc funds scholonto	http://semanticweb.kmi.open.ac.uk/ont...	
kmi successful at iswc 2004	http://semanticweb.kmi.open.ac.uk/ont...	
kmi successful at iswc 2004	http://semanticweb.kmi.open.ac.uk/ont...	
□a□ for akt	http://semanticweb.kmi.open.ac.uk/ont...	
kmi director chairs 14th international conference ...	http://semanticweb.kmi.open.ac.uk/ont...	
& pound;7m advanced knowledge technologies pr...	http://semanticweb.kmi.open.ac.uk/ont...	
bulgaria, germany, japan: the irs-iii tour continues	http://semanticweb.kmi.open.ac.uk/ont...	
climate predictions for 'the day after tomorrow'	http://semanticweb.kmi.open.ac.uk/ont...	
⊟ paper-in-proceedings		
supporting ontology driven document enrichment ...	http://semanticweb.kmi.open.ac.uk/ont...	
modelling data-intensive web sites with ontoweaver	http://semanticweb.kmi.open.ac.uk/ont...	
design of customized web applications with ontow...	http://semanticweb.kmi.open.ac.uk/ont...	
compendium: making meetings into knowledge ev...	http://semanticweb.kmi.open.ac.uk/ont...	
irs-ii: a framework and infrastructure for semantic...	http://semanticweb.kmi.open.ac.uk/ont...	
irsii: a framework and infrastructure for semantic ...	http://semanticweb.kmi.open.ac.uk/ont...	
template-driven information extraction for popula...	http://semanticweb.kmi.open.ac.uk/ont...	
iips: an intelligent information presentation system	http://semanticweb.kmi.open.ac.uk/ont...	
magpie towards a semantic web browser	http://semanticweb.kmi.open.ac.uk/ont...	
supporting ontology driven document enrichment ...	http://semanticweb.kmi.open.ac.uk/ont...	
an ontology-driven approach to web site generati...	http://semanticweb.kmi.open.ac.uk/ont...	
semantically marked up... now what?	http://semanticweb.kmi.open.ac.uk/ont...	
an ontology-driven approach to web site generati...	http://semanticweb.kmi.open.ac.uk/ont...	

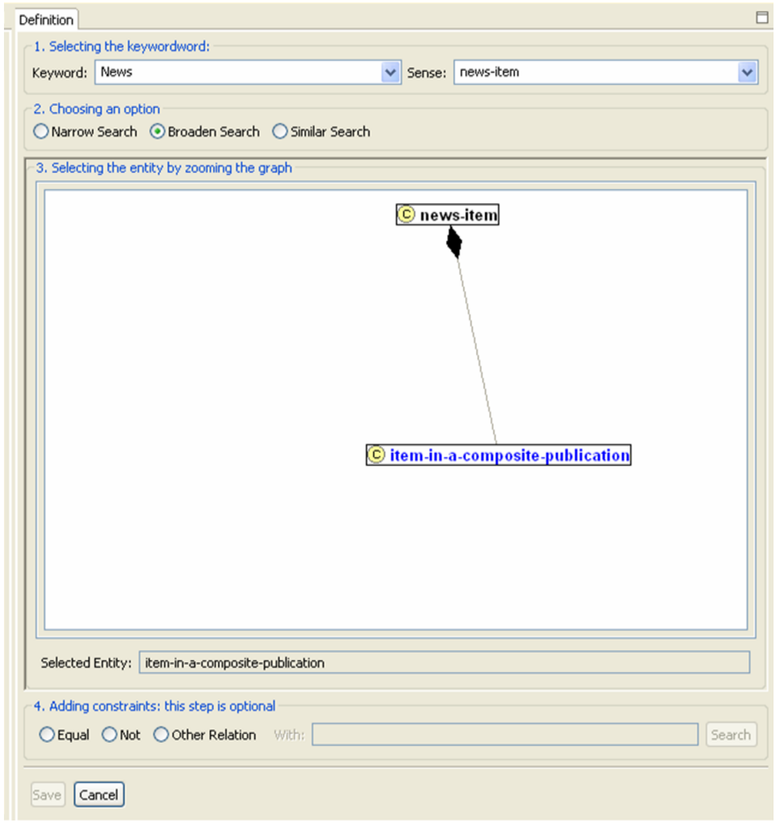
Fig. 10. The clustered results of the relation search example *"John:Enrico"*.

results, including tree-based views and graph-based views. We used ClusterMap[k] to generate graph-based views.

More advanced query refinement is supported by the concept of *knowledge lenses*, which supports a zoom-like facility in exploring the search space. It supports narrowing the search focus by going down to more specific classes (or topics) or broadening the search focus by going up to more generic classes. Figure 11 shows a screen snapshot of the user interface for refinement specification. Consider the query example "News:PhD students". The user can broaden the search from the class *news-item* (which describes news stories) to a more generic class *item-in-composite-publication*, which covers not only news stories but also other publications.

As shown in Figure 11, users can also add constraints to the search. For example, users can add a constraint "related to Semantic Web" to the query example described above. As the term *Semantic Web* matches the instance *semantic-web* whose type is the class *research-area*, the results will be news stories that are relevant to both PhD students and the research area Semantic Web.

[k]http://www.aduna-software.org/home/overview.view


Definition
1. Selecting the keywordword:
2. Choosing an option
3. Selecting the entity by zooming the graph
news-item
item-in-a-composite-publication
4. Adding constraints: this step is optional
Equal
Not
Other Relation
With:
Search
Save
Cancel

Keyword: News
Sense: news-item
Narrow Search [✓ Broaden Search] Similar Search
Selected Entity: item-in-a-composite-publication

Fig. 11. A screen snapshot of the user interface for refinement specification.

To summarize, SemSearch improves end user support with respect to the specification of complex queries by several means:

- *A Google-like query interface*, which offers a straightforward way to specifying relatively complex queries, where multiple keywords are involved.
- *Keyword based search as a starting point*, which offers users an easy way to start complex queries.
- *Multi-keyword search facility*, which supports relation search as well as entity search.
- *Knowledge lenses for query refinement*, which provide an intuitive way to broaden or narrow the search scope.
- *An intuitive and easy to use interface*, which supports all the interactions required to help users formulate and refine their queries.

7 SemBrowser: Browsing Networks of Entities

The key role of SemBrowser is to support users to browse and navigate through the knowledge network of the problem domain. Central to SemBrowser is *a relation extraction engine*, which, on the one hand, generates navigation services for a given entity, thus enabling navigation, and on the other hand, is responsible for retrieving entities and relations that satisfy the specified query requirements.

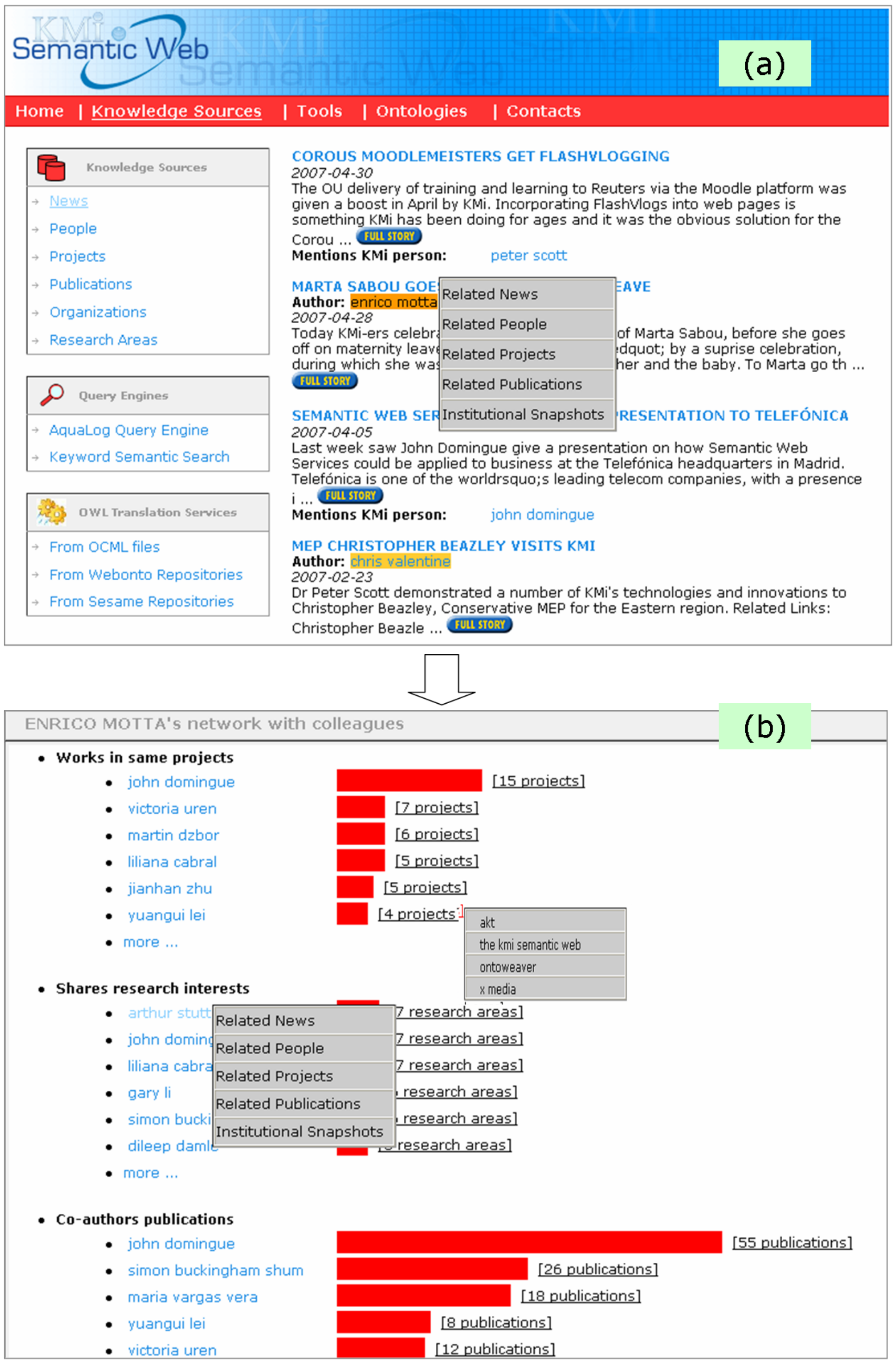
The purpose of navigation services is to narrow the scope of the knowledge network of the associated entity from containing all the relations to its neighborhood entities down to the specified relation only. To generate navigation services, the relation extraction engine first investigates direct relations, i.e., relations that are physically stored in the metadata repositories. For example, in the case of generating navigation services for Enrico Motta, the extraction first looks for triples in the format of *(X,relation, Enrico Motta)* or *(Enrico Motta, relation, X)* that are physically stored in the repositories. Such an operation yields several results in the KMi context, including news, publications, projects, etc. The relation extraction engine then walks around the graph of the given entity to collect indirect relations, through which the given entity is linked with other entities. The navigation service *"Related People"* is such an example for a specific person in the KMi context, as there are no triples in the metadata repositories that take the format *(people, relation, people).* SemBrowser makes use of the relation search approach described in Section 6 to find indirect relations and entities that are connected by these relations.

In the KMi context, several navigation services can be automatically extracted for a person, including *related news*, *related persons*, *related projects* and *related publications*. Such services are attached to semantic entities (which are people in the example) in Web pages as right click menus to enable navigation. Figure 12 shows an example of such navigation services. Each entity in the Web page is clickable and is associated with a set of navigation services. By clicking on the "Related People" menu associated with the person *Enrico Motta* (as shown in part (a)), the user can browse people who collaborate with Enrico Motta.

Behind the scenes, SemBrowser issues queries to the underlying metadata repository and gathers related entities. These entities are again associated with navigation services, as shown in part (b) of the figure. In this way, entity navigation is enabled.

Constraints can be added to the navigation services to filter out unwanted entities and their corresponding relations, thus enabling the customization of knowledge networks of an entity. For example, in the KMi SW portal context, we assume that users are only interested in certain types of relations when viewing the knowledge network of a specific person and his/her colleagues, including co-authored publications, shared research interests, and projects they collaborated in. Such constraints are added to the navigation service "Related People" and take effect during navigation.

Apart from the relation extraction engine, SemBrowser also provides *a visualization engine*, which presents knowledge networks and enables navigation. Two main techniques are used. One is an HTML-based approach, which uses the length of bars to visualize relation strength. A simple algorithm is developed, which assigns different weights to the extracted relations according to their path length to the given entity, and calculates the strength of the relations between two given entities. A heuristic rule used here is that the longer the path is, the weaker the relation gets. Figure 12(b) visualizes the relations between Enrico Motta and his colleagues using this technique. It provides a straightforward way of highlighting the entities that are closely related. From the figure, the user can easily determine that Enrico Motta has lots of collaboration with John Domingue.



COROUS MOODLEMEISTERS GET FLASHVLOGGING
2007-04-30
The OU delivery of training and learning to Reuters via the Moodle platform was given a boost in April by KMi. Incorporating FlashVlogs into web pages is something KMi has been doing for ages and it was the obvious solution for the Corou ...
Mentions KMi person: peter scott

MARTA SABOU GOE ... EAVE
Author: enrico motta
2007-04-28
Today KMi-ers celebr ... of Marta Sabou, before she goes off on maternity leave ... dquot; by a suprise celebration, during which she was ... her and the baby. To Marta go th ...

SEMANTIC WEB SER ... RESENTATION TO TELEFÓNICA
2007-04-05
Last week saw John Domingue give a presentation on how Semantic Web Services could be applied to business at the Telefónica headquarters in Madrid. Telefónica is one of the worldrsquo;s leading telecom companies, with a presence i ...
Mentions KMi person: john domingue

MEP CHRISTOPHER BEAZLEY VISITS KMI
Author: chris valentine
2007-02-23
Dr Peter Scott demonstrated a number of KMi's technologies and innovations to Christopher Beazley, Conservative MEP for the Eastern region. Related Links: Christopher Beazle ...

ENRICO MOTTA's network with colleagues

- **Works in same projects**
  - john domingue [15 projects]
  - victoria uren [7 projects]
  - martin dzbor [6 projects]
  - liliana cabral [5 projects]
  - jianhan zhu [5 projects]
  - yuangui lei [4 projects]
  - more ...
- **Shares research interests**
  - arthur stutt ... [7 research areas]
  - john doming ... [7 research areas]
  - liliana cabra ... [7 research areas]
  - gary li ... research areas]
  - simon bucki ... research areas]
  - dileep daml ... research areas]
  - more ...
- **Co-authors publications**
  - john domingue [55 publications]
  - simon buckingham shum [26 publications]
  - maria vargas vera [18 publications]
  - yuangui lei [8 publications]
  - victoria uren [12 publications]

Fig. 12. An example of entity navigation in the context of the KMi SW portal. Part (a) shows sample navigation services. Part (b) displays the network of Enrico Motta with his colleagues. The user can navigate to (b) by clicking on the right-click menu attached with semantic entities as shown in (a).

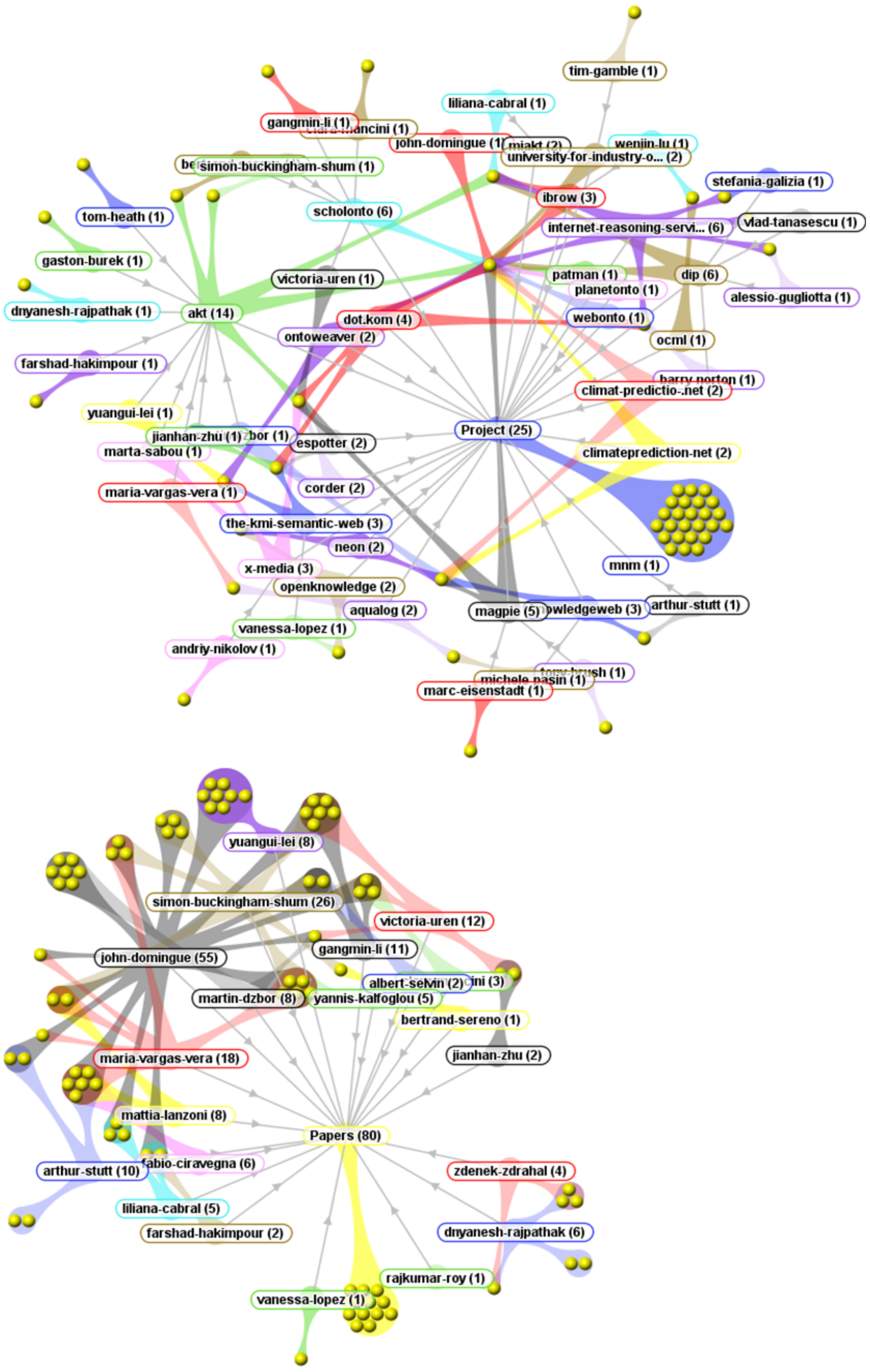


Fig. 13. A sample knowledge network visualized in ClusterMap.

The other main technique used is a graph-based approach, which uses graphs to represent semantic relations. We use ClusterMap for this purpose. Figure 13 shows a screenshot of the example knowledge network described above. Apart from listing all the entities that are around the given entity, the visualization engine also shows the entities that are linked to these entities. It shows Enrico Motta works on 25 projects and published 80 papers. Furthermore, it shows the network of people's collaboration around the projects and papers.

8 Related Work

As this work involves research in several different topics, we will briefly describe the related work respective to each.

8.1 Related work on semantic metadata acquisition

A number of tools have been developed recently, which support the acquisition of semantic metadata. Some focus on providing support for manual operations. Some aim to provide fully automatic or semi-automatic support. The closest tools are those that support metadata extraction from structured data sources (e.g., database and XML feeds) and semi-structured (e.g. Web pages, tables). Examples include the Semantic Content Organization and Retrieval Engine (SCORE) [21], the KIM platform [22], and the SW portals described in Section 2. As a way of ensuring high quality, domain specific rules are often used to specify how and where to extract specific information. Our work distinguishes from these efforts by providing more comprehensive support for quality control.

As described in Section 4, ASDI provides several means to ensure the quality of the extracted data. First, it aims to reduce ambiguities by taking into account the context in which an entity is mentioned in order to determine its type. Second, it contains a verification engine that checks the validity of any derived metadata against a repository of trusted domain knowledge and against the information available on the Web. Finally, since the whole acquisition process is automatic, it can be automatically run whenever new data becomes available, thus ensuring that the semantic metadata is always up to date.

8.2 Related work on semantic search

Numerous search tools have been recently developed which aim to enhance the performance of traditional search technologies by exploiting semantic metadata. Among them, ORAKEL [15] and e-librarian [23] are the closest tools to our question answering tool AquaLog. Like AquaLog, they aim to exploit the domain ontology and the available semantic metadata to answer questions submitted in natural language format. While AquaLog is a portable tool, which can be easily adapted to different domains, the portability of ORAKEL and e-librarian is reduced in order to obtain high performance. For instance, ORAKEL involves the user in the tasks of providing a domain specific grammar and lexicon, while the e-librarian requires a domain dictionary to be built.

Ontogator [24], Corese [25], and the DOSE search engine [26] are closest to SemSearch. Like SemSearch, which uses keyword based search as a starting point for complex queries, both Ontogator and Corese combine keyword search with the other searching facilities to speed up the search process. The DOSE search engine, on the other hand, is similar to our search tool in using ontology navigation as a way to support query refinement. GRQL [27] and SEWASIE

[28] are also notable tools that use graph-based views to support query construction.

Regarding keyword based search, the closest work includes the TAP search engine [19] and the search engine presented in [20]. While these search tools make use of graph walking to retrieve relevant entities, SemSearch focuses on translating user keywords into formal queries.

As to relation search, TEXTRUNNER [29] is the closest tool to our work. Just like SemSearch, TEXTRUNNER accepts several query terms as input and produces entities or relations as output depending on the type of the search. One major difference between the two lies in the knowledge graphs they work on: TEXTRUNNER with string-based graphs and SemSearch with knowledge-based graphs (i.e., semantic networks). The approach presented by Anyanwu & Sheth [30] shares a similar goal with our work, which is to support the retrieving of semantic associations between different data entities. A query language is proposed, which relies on a set of query operators to enable the specification of relation based queries. Another tool which is worth mentioning here is the tool developed by Li et al. [31]. It emphasis the importance of the relations between keywords in document retrieving.

8.3 Related work on semantic browsing

Notable tools for semantic browsing are Magpie [32] and Haystack [33]. Magpie attaches the available semantic metadata to existing Web pages and extracts several services from the metadata to enable semantic browsing. Thus, it is very different from our approach, which extracts semantic metadata from Web pages and makes use of the extracted semantic metadata to support semantic browsing.

Haystack, on the other hand, takes a route similar to ours to approach semantic browsing. While Haystack focuses on how to support end users in defining user interfaces to organize the information space of the problem domain, SemBrowser concentrates on how to generate and customize navigation services for a given entity to meet users' information seeking needs.

Other work which is worth mentioning here includes the tools developed in the research strand of Semantic Wikipedia [34]. They aim to enhance knowledge sharing and exchange fostered by the Wiki technology by using semantic metadata to manage and store Wiki pages and their links.

9 Conclusions and Future Work

The key contribution of this paper is the proposed portal infrastructure, which integrates and exploits several comprehensive functionalities that are essential in enhancing access to knowledge in traditional Web portals and state-of-art Semantic Web portals, namely high quality metadata acquisition, semantic search, and semantic browsing.

Central to the infrastructure are four distinctive tools: i) *ASDI*, which supports metadata extraction from heterogenous data sources and provides several means to address quality control; ii) *AquaLog*, which exploits the domain ontology and natural language processing techniques to answer queries in natural language format, thus lowering barriers for users to make semantic queries; iii) *SemSearch*, which addresses two important issues associated with semantic search namely relation search and query specification; and iv) *SemBrowser*, which supports the browsing of knowledge networks of individual entities.

While these tools can each work separately, the integration is able to bring knowledge contained in the underlying data sources to ordinary end users in the context of Semantic

Web portals, that would not be possible otherwise. We have built a Semantic Web portal for our research lab by applying the proposed portal infrastructure. Experimental evaluations have been carried out to assess the performance of the metadata extraction tool ASDI and question answering tool AquaLog.

With 91% precision and 77% recall, ASDI improves the quality of the metadata that are extracted by the information extraction tool. This indicates that the quality control measures developed take effect. As to AquaLog, we collected 69 different questions. Among them, 40 have been handled correctly. 19 more can be handled correctly if re-formatted by the 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). Please refer to [9] and [10] for the details about the evaluation set up and the analysis of results.

The presented infrastructure also addresses the heterogeneity issue of SW portals at the data source level. As described in Section 4.1, it relies on the metadata extraction component, ASDI, to address semi-structured as well as structured data sources. Our future work will focus on the heterogeneity issue at the ontology level. This calls for extending the tool suite towards multiple ontologies. In particular, measures need to be developed to compare and combine knowledge represented in different ontologies in order to enable quality control, semantic search, and semantic browsing.

Acknowledgement

We wish to thank Dr. Marta Sabou for her 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. N. Stojanovic, A. Maedche, S. Staab, and R. Studer (2001), *SEAL - a framework for developing SEmantic PortALs*, In Proceedings of the 1st International Conference on Knowledge Capture(KCAP 2001), pp. 155–162.
2. Y. Jin, S. Decker, and G. Wiederhold (2001), *OntoWebber: model-driven ontology-based Web site management*, In Proceedings of Semantic Web Working Symposium (SWWS), pp. 529–547.
3. O. Corcho, A. Gomez-Perez, A. Lopez-Cima, V. Lopez-Garcia, and M. C. Suarez-Figueroa (2003), *ODESeW. Automatic generation of knowledge portals for Intranets and Extranets*, In Proceedings of the 2nd International Semantic Web Conference 2003 (ISWC 2003), pp. 20–23.
4. M.C. Schraefel, N.R. Shadbolt, N. Gibbins, H. Glaser, and S. Harris (2004), *CS AKTive Space: representing computer science in the Semantic Web*, In Proceedings of the 13th International World Wide Web Conference (WWW 2004), pp. 384–392.
5. P. Mika (2005), *Flink: Semantic Web technology for the extraction and analysis of social networks*, Journal of Web Semantics, 3(2-3):211–223.
6. E. Hyvonen, E. Makela, M. Salminen, A. Valo, K. Viljanen, S. Saarela, M. Junnila, and S. Kettula (2005), *MuseumFinland – Finnish museums on the Semantic Web*, Journal of Web Semantics, 3(2-3):224–241.

7. M. Schraefel, M. Wilson, A. Russell, and D. Smith (2006), *mSpace: improving information access to multimedia domains with multimodal exploratory search*, Commun. ACM, 49(4):47–49.
8. G. Schreiber, A. Amin, M. van Assem, V. de Boer, L. Hardman, M. Hildebrand, L. Hollink, Z. Huang, J. van Kersen, M. de Niet, B. Omelayenko, J. van Ossenbruggen, R. Siebes, J. Taekema, J. Wielemaker, and B. Wielinga (2006), *MultimediaN E-Culture demonstrator*, In Proceedings of the 5th International Semantic Web Conference (ISWC 2006), pp. 951–958.
9. Y. Lei, M. Sabou, V. Lopez, J. Zhu, V. S. Uren, and E. Motta (2006), *An infrastructure for acquiring high quality semantic metadata*, In Proceedings of the 3rd European Semantic Web Conference, pp. 230–244.
10. V. Lopez, E. Motta, V. Uren, and M. Pasin (2007), *AquaLog: an ontology-driven question answering system as an interface to the Semantic Web*, Jounal of Web Semantics: Science, Services and Agents on the World Wide Web, 5(2):72–105.
11. Y. Lei, V. Uren, and E.Motta (2006), *SemSearch: a search engine for the Semantic Web*, In Proceedings of the 14th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2006), pp. 238–245.
12. Y. Lei, E. Motta, and J. Domingue (2004), *OntoWeaver-S: supporting the design of knowledge portals*, In Proceedings of the 14th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2004), pp. 216–230.
13. J. Zhu, V. Uren, and E. Motta (2004), *ESpotter: adaptive named entity recognition for web browsing*, In Proceedings of the Professional Knowledge Management Conference, pp. 518–529.
14. Y. Lei (2005), *An instance mapping ontology for the Semantic Web*, In Proceedings of the 3rd International Conference on Knowledge Capture, pp. 67–74.
15. P. Cimiano, S. Handschuh, and S. Staab (2004), *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 (WWW 2004), pp. 462–471.
16. C. Fellbaum (1998), *WORDNET: An electronic lexical database.* MIT Press.
17. H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan (2002), *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 2002), pp. 168–175.
18. W. Cohen, P. Ravikumar, and S. Fienberg (2003), *A comparison of string distance metrics for name-matching tasks*, In Proceedings of the IJCAI-2003 Workshop on Information Integration on the Web (IIWeb-03), pp. 73–78.
19. R. Guha, R. McCool, and E. Miller (2003), *Semantic search*, In Proceedings of the 12th International Conference on World Wide Web, pp. 700–709.
20. C. Rocha, D. Schwabe, and M. de Aragao (2004), *A hybrid approach for searching in the Semantic Web*, In Proceedings of the 13th International World Wide Web Conference (WWW 2004), pp. 374–383.
21. A. Sheth, C. Bertram, D. Avant, B. Hammond, K. Kochut, and Y. Warke (2002), *Semantic Content Management for Enterprises and the Web*, IEEE Internet Computing, 6(44):80–87.
22. B. Popov, A. Kiryakov, A. Kirilov, D. Manov, D. Ognyanoff, and M. Goranov (2003), *KIM - semantic annotation platform*, In Proceedings of the 2nd International Semantic Web Conference (ISWC 2003), pp. 834–849.
23. S. Linckels (2005), *A simple solution for an intelligent librarian system*, In Proceedings of the IADIS International Conference of Applied Computing, pp. 495–503.
24. E. Makela, E. Hyvonen, and S. Saarela (2006), *Ontogator: a semantic view-based search engine service for web applications*, In Proceedings of the 5th International Semantic Web Conference (ISWC 2006), pp. 92–106.
25. O. Corby, R. Dieng-Kuntz, and C. Faron-Zucker (2004), *Querying the Semantic Web with Corese search engine*, In Proceedings of the 3rd Prestigious Applications Intelligent Systems Conference (PAIS) on 16th European Conference on Artificial Intelligence (ECAI/PAIS), pp. 705–709.
26. D. Bonino, L. Farinetti, and A. Bosca (2004), *Ontology driven semantic search*, WSEAS Transaction on Information Science and Application, 1(6):1597–1605.

27. N. Athanasis, V. Christophides, and D. Kotzinos (2004), *Generating on the fly queries for the Semantic Web: The ICS-FORTH graphical RQL interface (GRQL)*, In Proceedings of the 3rd International Semantic Web Conference (ISWC 2004), pp. 486–501.
28. T. Catarci, T. Di Mascio, E. Franconi, G. Santucci, and S. Tessaris (2004), *An ontology based visual tool for query formulation support*, In Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 2004), pp. 32–33.
29. M. Cafarella, M. Banko, and O. Etzioni (2006), *Relational web search*, Technical Report UW-CSE-2006-04-02, Turing Center, University of Washington.
30. K. Anyanwu and A. Sheth (2003), *p-Queries: enabling querying for semantic associations on the Semantic Web*, In Proceedings of the 12th International Conference on World Wide Web (WWW 2003), pp. 690–699.
31. Y. Li, Y. Wang, and X. Huang (2007), *A relation-based search engine in Semantic Web*, IEEE Transactions on Knowledge and Data Engineering, 19(2):273–282.
32. J. B. Domingue and M. Dzbor (2004), *Magpie: browsing and navigating on the Semantic Web*, In Proceedings of the 9th International Conference on Intelligent User Interfaces (IUI 2004), pp. 191–197.
33. D. Quan, D. Huynh, and D. Karger (2003), *Haystack: a platform for authoring end user semantic web applications*, In Proceedings of the 2nd International Semantic Web Conference 2003 (ISWC 2003), pp. 738–753.
34. M. Vlkel, M. Krtzsch, D. Vrandecic, H. Haller, and R. Studer (2006), *Semantic MediaWiki*, In Proceedings of the 15th International Conference on World Wide Web (WWW 2006), pp. 585–594.