Ontology-driven Question Answering in AquaLog

Vanessa Lopez, Enrico Motta
Knowledge Media Institute, The Open University.
Walton Hall, Milton Keynes,
MK7 6AA, United Kingdom.
{v.lopez, e.motta}@open.ac.uk

Motivation for AquaLog

The *semantic web* vision is one in which rich, *ontology-based semantic markup* is widely available, both to enable sophisticated interoperability among agents and to support human web users in locating and making sense of information. The availability of semantic markup on the web also opens the way to novel, sophisticated forms of question answering. While semantic information can be used in several different ways to improve question answering, an important (and fairly obvious) consequence of the availability of semantic markup on the web is that this can indeed be queried directly. For instance, we are currently augmenting our departmental web site, http://kmi.open.ac.uk, with semantic markup, by instantiating an ontology describing academic life [4] with information about our personnel, projects, technologies, events, etc., which is automatically extracted from departmental databases and unstructured web pages. In the context of standard, keyword-based search this semantic markup makes it possible to ensure that standard search queries, such as “peter scott home page kmi”, actually return Dr Peter Scott’s home page as their first result, rather than some other resource (as indeed is the case when using current non-semantic search engines on this particular query). Moreover, as pointed out above, we can also query this semantic markup directly. For instance, we can ask a query such as “what are the projects in the semantic web area” and, thanks to an inference engine able to reason about the semantic markup and draw inferences from axioms in the ontology, we can then get the correct answer.
Hence, in the first instance, the work on the AquaLog query answering system is based on the premise that the semantic web will benefit from the availability of natural language query interfaces, which allow users to query semantic markup viewed as a knowledge base. We believe that in the semantic web scenario it makes sense to provide query answering systems on the semantic web, *which are portable with respect to ontologies*, our AquaLog system allows to choose an ontology and then ask queries with respect to the universe of discourse covered by the ontology.

The AquaLog Approach

Hence, AquaLog is our portable question-answering system which takes queries expressed in natural language and an ontology as input and returns answers drawn from one or more knowledge bases (KBs), which instantiate the input ontology with domain-specific information. AquaLog makes use of the GATE NLP platform, string metrics algorithms, WordNet and a novel ontology-based *relation similarity service* to make sense of user queries with respect to the target knowledge base. Also, AquaLog is couple with a portable and contextualized learning mechanism to obtain domain-dependent knowledge by creating a lexicon.
AquaLog divides the task of mapping user queries to answers into two main subtasks: the linguistic component produce an intermediate logical representation (which subscribes to a binary relational model like RDF-based knowledge representation (KR) formalisms for the semantic web) from the input, afterwards, this intermediate representation is further processed by the Relation Similarity Service, responsible of mapping this intermediate query into a form consistent with the target knowledge base.
A key feature of the Linguistic Component is the dynamic classification of queries through the use of regular expressions over annotations, through the use of gate. The current version of AquaLog identifies around 23 different linguistic categories, depending not only upon the kind of solution that needs to be achieved, but also gives an indication to the Linguistic Component about how to create the intermediate representation.

The AquaLog current implementation

It is implemented in Java as a web application, using a client-server architecture[1]. Although AquaLog has primarily been designed for use with semantic web languages, it makes use of a generic plug-in mechanism, which means it can be easily interfaced to different ontology servers and knowledge representation platforms.

[1] AquaLog system can be accessed in: http://plainmoor.open.ac.uk:8080/JavaAQUAv1.0

Currently we use it with our own OCML-based KR infrastructure, although in the future we plan to provide direct plug-in mechanisms for use with the emerging RDF and OWL servers.
There is not a single strategy here, the novel Relation Similarity Service –RSS- essentially it tries to make sense of the input query by looking at the structure of the ontology and the information stored in the target KBs. A typical situation the RSS has to cope with is one in which the structure of the intermediate query does not match the way the information is represented in the ontology. For instance, the query "who is the secretary in kmi?", as shown if figure xxx, may be parsed into <person, secretary, kmi>, following a pure linguistic criteria, while the ontology may be organized in terms of <secretary, works-for, kmi>. Also in these cases the RSS is able to reason about the mismatch, re-classify the intermediate query and generate the correct logical query. The example is shown on the left screen in Figure 1, also a second example is shown on the right screen.

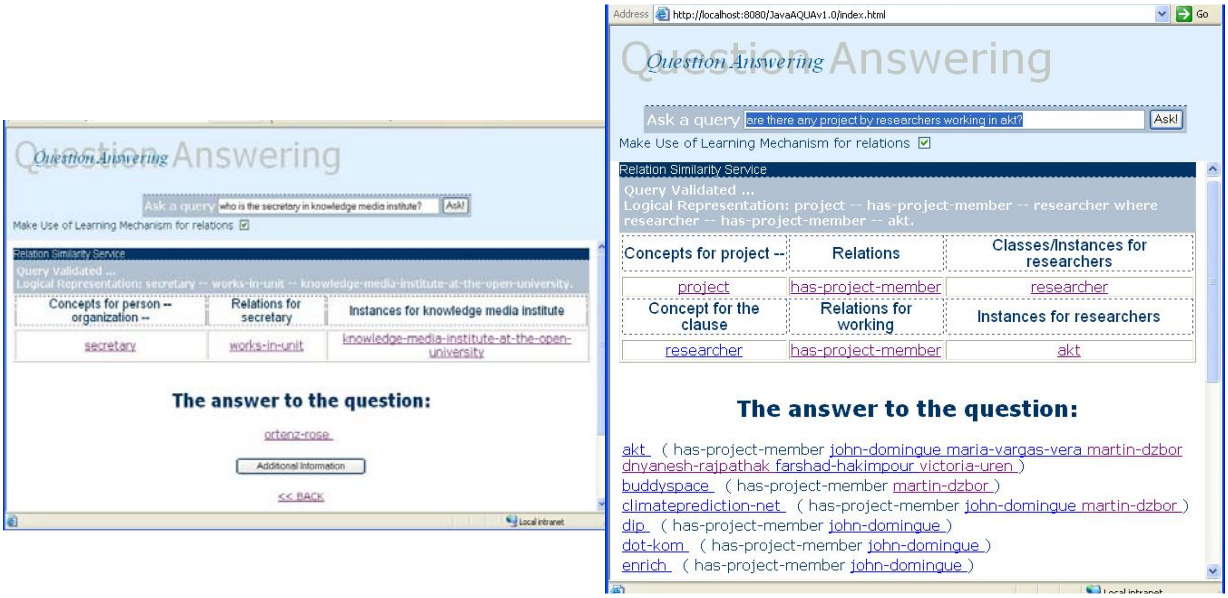

Figure 1. Example of AquaLog in action

An important aspect of the RSS is that it is interactive. In other words when unsure it will ask the user for help, e.g., when it is not able to disambiguate or interpret the query. It is important to emphasize that calling on the users to participate in the QA process is only done if no information is available to AquaLog to disambiguate the query directly. For instance let's consider the two queries shown in Figure 2. On the right screen we are looking for the web address of Peter and given that the system is unable to disambiguate between Peter-Scott, Peter-Sharpe or Peter-Whalley, user's feedback is required. However, on the left screen we are asking for the web address of Peter, who has an interest in knowledge reuse. In this case AquaLog does not need assistance from the user, given that only one of the three Peters has an interest in knowledge reuse.

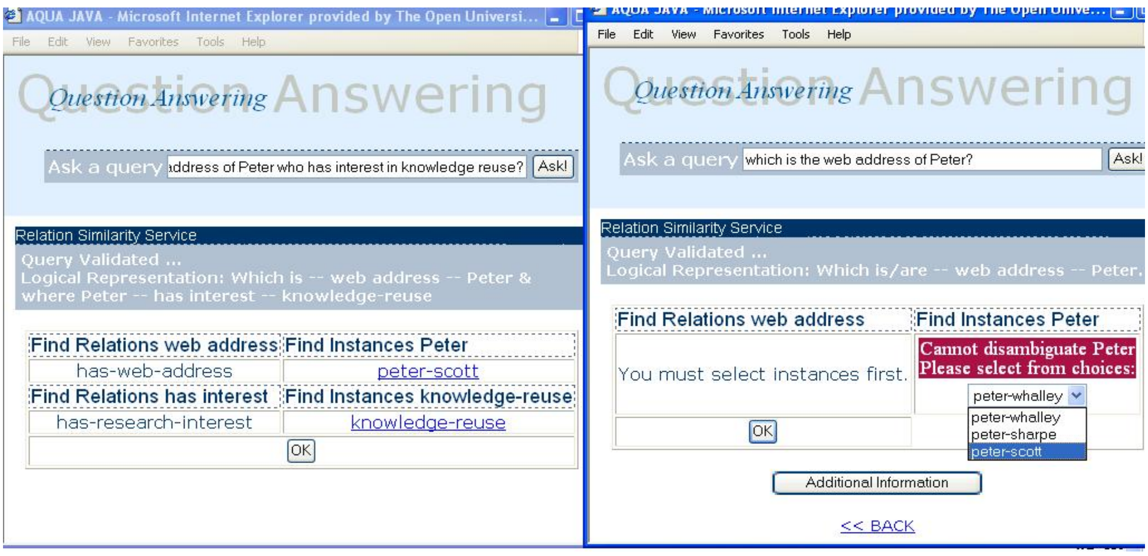

Figure 13. Example of user-driven and automatic disambiguation