KNOWLEDGE MEDIA

KMi

INSTITUTE

Ontology-driven Question Answering In AquaLog

Tech Report kmi-04-5
April 2004

Vanessa Lopez and Enrico Motta

The Open University

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

Abstract 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. AquaLog is a 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. Finally, 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 Introduction

The *semantic web* vision [1]is one in which rich, *ontology-based semantic markup* is widely available, both to enable sophisticated interoperability among agents, e.g., in the e-commerce area, and to support human web users in locating and making sense of information. For instance, tools such as Magpie [2] support sense-making and semantic web browsing by allowing users to select a particular ontology and use it as a kind of 'semantic lens', which assists them in making sense of the information they are looking at. As discussed by McGuinness in her recent essay on "Question Answering on the Semantic Web" [3], the availability of semantic markup on the web also opens the way to novel, sophisticated forms of question answering, which not only can potentially provide increased precision and recall compared to today's search engines, but are also capable of offering additional functionalities, such as i) proactively offering additional information about an answer, ii) providing measures of reliability and trust and/or iii) explaining how the answer was derived.

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 "list all the kmi 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.

This scenario is of course very similar to asking natural language queries to databases (NLDB), which has long been an area of research in the artificial intelligence and database communities [5] [6] [7] [8] [9], even if in the past decade has somewhat gone out of fashion [10] [11]. However, it is our view that the semantic web provides a new and potentially very important context in which results from this area of research can be applied. Moreover, interestingly from a research point of view, it provides a new 'twist' on the old issues associated with NLDB research. Hence, in the first instance, the work on the AquaLog query answering system described in this paper 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. Moreover, similarly to the approach we have adopted in the Magpie system, 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*. In other words, just like in the case of Magpie, where the user is able to select an ontology (essentially a semantic viewpoint) and then browse the web through this semantic filter, our AquaLog system allows the user to choose an ontology and then ask queries with respect to the universe of discourse covered by the ontology.

2 The AquaLog Approach

AquaLog is a 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. As already emphasized, a key feature of AquaLog is that it is modular with respect to the input ontology, the aim here being that it should be zero cost to switch from one ontology to another when using AquaLog.

AquaLog is part of the AQUA [12] framework for question answering on the semantic web and in particular addresses the upstream part of the AQUA process, the translation of NL queries into logical ones and the interpretation of these NL-derived logical queries with respect to a given ontology and available semantic markup.

AquaLog adopts a *triple-based* data model and, as shown in figure 1, the role of the linguistic component of AquaLog is to translate the input query into a set of intermediate triples, of the form <subject, predicate, object>. These are then further processed by a module called the "Relation Similarity Service" (RSS), to produce ontology-compliant queries. For example, in the context of the academic domain mentioned earlier, AquaLog is able to translate the question "Who is a Professor at the Knowledge Media Institute?" into the following, ontology-compliant logical query, <typeOf ?x Professor-in-Academia> & <works-in-unit ?x KMi>, expressed as a conjunction of non-ground triples (i.e., triples containing variables). The role of the RSS is to map the intermediate form, <?who, Professor, KMi> into the target, ontology-compliant query.

There are two main reasons for adopting a triple-based data model. First of all, as Katz et al. point out [13], although not all possible queries can be represented in the binary relational model, in practice these occur very frequently. Secondly, RDF-based knowledge representation (KR) formalisms for the semantic web, such as RDF itself [14] or OWL [15] also sub-

scribe to this binary relational model and express statements as <subject, predicate, object>. Hence, it makes sense for a query system targeted at the semantic web to adopt this data model.

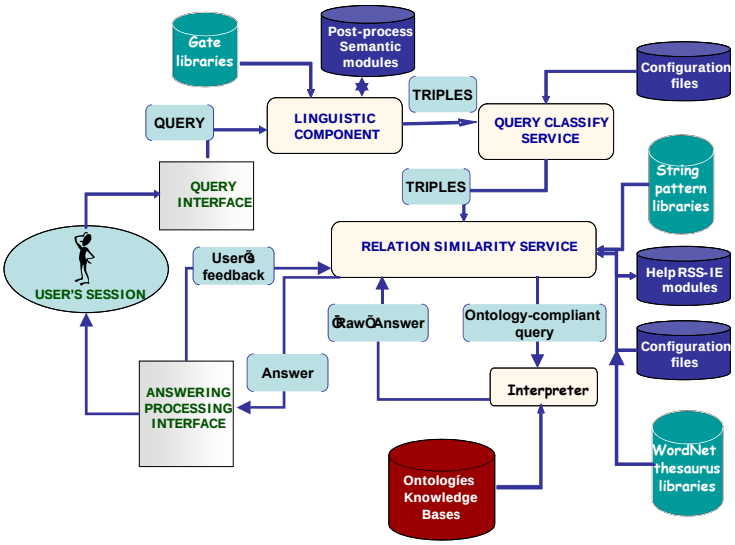

Figure 1. The Architecture of AquaLog

We have seen that, in common with most other NLDB systems, AquaLog divides the task of mapping user queries to answers into two main subtasks: producing an intermediate logical representation from the input query and mapping this intermediate query into a form consistent with the target knowledge base. Moreover it explicitly restricts the range of questions the user is allowed to ask to a set of expressions/syntactic patterns, so that the linguistic limits of the system are obvious to the user (to avoid the effort of rephrasing questions) and to ensure users understand whether a query to AquaLog failed for reasons which are *linguistic* (failure to understand the linguistic structure of the question) or *conceptual* (failure of the ontology to provide meaning to the concepts in the query)**.**

In the next section we describe the AquaLog architecture in more detail.

3 The AquaLog Architecture

AquaLog was designed with the aim of making the system as flexible and as modular as possible. It is implemented in Java as a web application, using a client-server architecture. A key feature is the use of a plug-in mechanism, which allows AquaLog to be configured for different KR languages. Currently we use it with our own OCML-based KR infrastructure [16] [17], although in the future we plan to provide direct plug-in mechanisms for use with the emerging RDF and OWL servers[1].

[1] In any case we are able to import and export RDF(S) and OWL from OCML, so the lack of an explicit RDF/OWL plug-in is actually not a problem in practice.

3.1 Initialization and user's session

At initialization time the AquaLog server will access and cache basic indexing data for the target KB(s), so that they can be efficiently accessed by the remote clients to guarantee real-time question answering, even when multiple users access the server simultaneously.

3.2 Gate framework for natural language & query classify service

AquaLog uses the GATE [18] [19] infrastructure and resources (language resources, processing resources like ANNIE, serial controllers, etc.) to map the input query in natural language to the triple-based data model. Communication between AquaLog and GATE takes place through the standard GATE API. The GATE chunker used for this task does not actually generate a parse tree. As discussed by Katz et al. [20] [21], although parse trees (as for example, the NLP parser for Stanford [22]) capture syntactic relations, they are often time-consuming and difficult to manipulate. Moreover, we also found that in our context we could do away with much of the parsed information. For the intermediate representation, we use the triple-based data model rather than logic, because at this stage we do not have to worry about getting the representation right. The role of the intermediate representation is simply to provide an easy to manipulate input for the RSS.

After the execution of the GATE controller a set of syntactical annotations are returned associated with the input query. Annotations include information about sentences, tokens, nouns and verbs. For example we get voice and tense for the verbs, or categories for the nouns, such as determinant, singular/plural, conjunction, possessive, determiner, preposition, existential, wh-determiner, etc. When developing AquaLog we extended the set of annotations returned by GATE, by identifying relations and question indicators (which/who/when/etc.). This was achieved through the use of *Jape grammars*. These consist of a set of *phases*, which run sequentially, each of which defined as a set of pattern rules. The reason for this extension was to be able to clearly identify the scope of a relation – e.g., to be able to identify "has research interest" as a relation. Here we exploited the fact that natural language commonly employs a preposition to express a relationship.

Although it is not necessary, we could improve the precision of the AquaLog linguistic component by modifying or creating the appropriate jape rules for specific cases. For example, the word "project" could be understood as a noun or as a verb, depending on the priority of the rules. Another example is when some disambiguation is necessary as in the example: "Has john research interest in ontologies?". Here "research" could be either the last name of John or a noun part of the relation "has research interest"[2].

As shown in figure 1, the architecture also includes a post-processing semantic (PPS) module to further process the annotations obtained from the extended GATE component. For example when processing the query "John has research interest in ontologies", the PPS ensures that the relation is identified as "has research interest". Other more complex cases are also dealt with.

Finally, before passing the intermediate triples to the RSS, AquaLog performs two additional checks. If it is not possible to transform the question into a term-relation form or the question is not recognized, further explanation is given to the user, to help him to reformulate the question. If the question is valid, then a Query Classify Service is invoked to deter-

[2] Of course a better way to express the query would be "Has John got a research interest in ontologies?", which can be parsed with no problems. However, in our experience these slightly ungrammatical queries are very common and it is our aim to produce a system robust enough to deal with many of them.

mine the type of the question, e.g., a "where" question, and pass this information on to the Relation Similarity Service.

3.3 The Relation Similarity Service

This is the backbone of the question-answering system. The RSS is called after the NL query has been transformed into a term-relation form and classified and it is the main component responsible for producing an ontology-compliant logical query.

Essentially the RSS tries to make sense of the input query by looking at the structure of the ontology and the information stored in the target KBs, as well as using string similarity matching and lexical resources, such as WordNet. There is not a single strategy here, so we will not attempt to give a comprehensive overview of all the possible ways in which a query can be interpreted. Rather, we will show one example, which is illustrative of the way the RSS works.

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 between two possible relations which can be used to interpret the query.

For example, let's consider a simple question like "*who works on the semantic web?*". Here, the first step for the RSS is to identify that "semantic web" is actually a "research area" in the target KB[3]. If a successful match is found, the problem becomes one of finding a relation which links a person (or an organization) to the semantic web area.

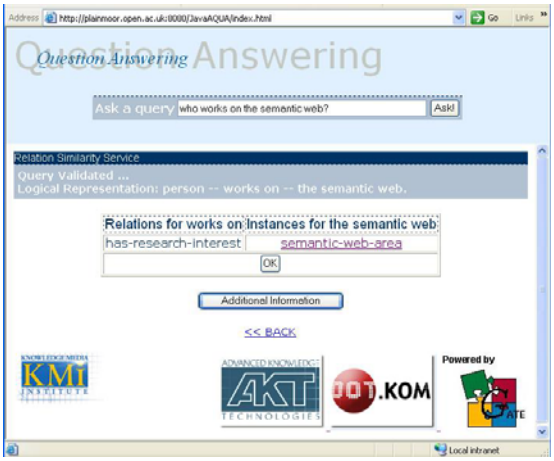

Figure 2. AquaLog in action.

By analyzing the KB, AquaLog finds that the only relation between a person and the semantic web area is has-research-interest and therefore suggests to the user that the question could be interpreted in terms of this relation. If the user clicks OK, then the answer to the query is provided. It is important to note that in order to make sense of the triple <person, works, semantic web>, all subclasses of person need to be considered, given that the relation has-research-interest could be defined only for researchers rather than people in general. If multiple relations are possible candidates for interpreting the query, then string matching is

[3] Naming convention vary depending on the KR used to represent the KB and may even change with ontologies – e.g., an ontology can have slots such as "variant name" or "pretty name". AquaLog deals with differences between KRs by means of the plug-in mechanism. Differences between ontologies need to be handled by specifying this information in a configuration file.

used to determine the most likely candidate, using the relation name, eventual aliases, or synonyms provided by lexical resources such as WordNet [23]. If no relations are found using this method, then the user is asked to choose from the current list of candidates. However, it is important to emphasise that calling on the users to disambiguate is only done if no information is available to AquaLog, which allows the system to disambiguate the query directly. For instance let's consider the two queries shown in figure 3. 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.

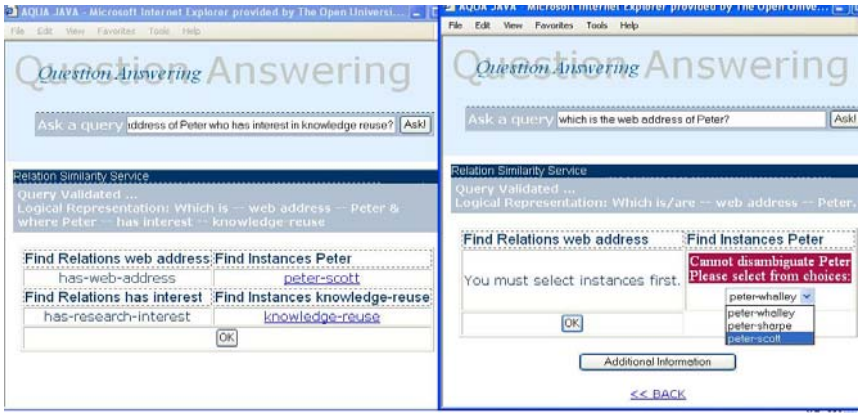

Figure 3. Automatic or user-driven disambiguation.

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 "which are the publications of John?" may be parsed into <John, publications, something>, while the ontology may be organized in terms of <Publication, has-author, Author>. Also in these cases the RSS is able to reason about the mismatch and generate the correct logical query.

3.4 Helping the user making sense of the information

We have already mentioned that AquaLog is an interactive system. If the RSS fails to make sense of a query, the user is asked to help choose the right relation or instance. Moreover, in order to help the user, AquaLog shows as much information as possible about the current query, including both information drawn from WordNet and from the ontology – see figure 4.

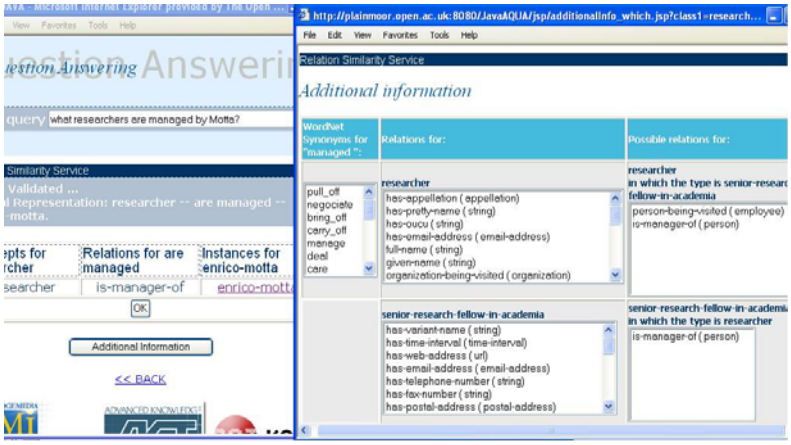

Figure 4. Providing additional information

Finally, when answers are presented to the user, it is also possible for him/her to use these answers as starting points for navigating the KB – see figure 5.

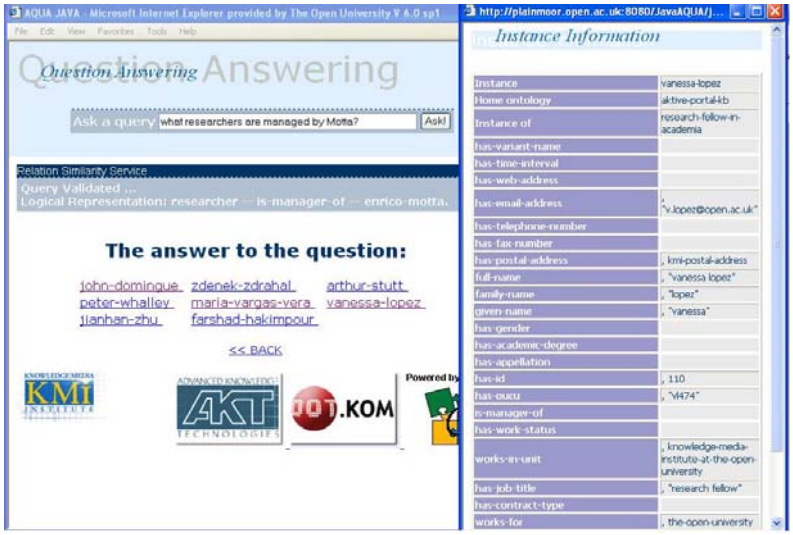

Figure 5. Navigating the KB starting from answers to queries.

3.5 String matching algorithms

String algorithms are used to find patterns in the ontology for any of the terms inside the intermediate triples obtained from the user's query. They are based on String Distance Metrics for Name-Matching Tasks, using an open-source from the Carnegie Mellon University in Pittsburgh [24]. This comprises a number of string distance metrics proposed by different communities, including edit-distance metrics, fast heuristic string comparators, token-based distance metrics, and hybrid methods. The conclusions of an experimental comparison of metrics realized by the University of Pittsburgh states: "Overall, the best-performing method is an hybrid scheme combining a TFIDF weighting scheme … with the Jaro-Winkler string distance scheme, developed in the probabilistic record linkage community".

However, it is not recommendable to trust just one metric. Experimental comparisons using the AKT ontology with different metrics show that the best performance is obtained with a combination of the following metrics: JaroWinkler, Level2JaroWinkler and Jaro.
A key aspect of using metrics is *thresholds*. By default two kinds of thresholds are used, called: "trustable" and "did you mean?", where the former is of course always preferable to the latter. AquaLog uses different thresholds depending on whether it is looking for concepts names, relations or instances.

4 Evaluation

4.1 Scenario and results

A full evaluation of AquaLog will require both an evaluation of its query answering ability as well an evaluation of the overall user experience. Moreover, because one of our key aims is to make AquaLog portable across ontologies, this aspect will also have to be evaluated formally. While a full evaluation has not been carried out yet, we performed an initial study, whose aim was to assess to what extent the AquaLog application built using AquaLog with the AKT ontology and the KMi knowledge base satisfied user expectations about the range of questions the system should be able to answer. A second aim of the experiment was also to provide information about the nature of the possible extensions needed to the ontology and the linguistic components – i.e., not only we wanted to assess the current coverage of the system but also get some data about the complexity of the possible changes required to generate the next version of the system. Thus, we asked ten members of KMi, none of whom has been involved in the AquaLog project, to generate questions for the system. Because one of the aims of the experiment was to measure the linguistic coverage of the system with respect to user needs, we did not provide them with any information about the linguistic ability of the system. However, we did tell them something about the conceptual coverage of the ontology, pointing out that its aim was to model the key elements of a research lab, such as publications, technologies, projects, research areas, people, etc.
We also pointed out that the current KB is limited in its handling of temporal information, therefore we asked them not to ask questions which required sophisticated temporal reasoning. Because no 'quality control' was carried out on the questions, it was perfectly OK for these to contain spelling mistakes and even grammatical errors.

We collected in total 76 different questions, 37 of which were handled correctly by AquaLog, i.e., 48.68% of the total. This was a pretty good result, considering that no linguistic restriction was imposed on the questions.

We analyzed the failures and divided them into the following five categories (the total adds up to more than 37 because a query may fail at several different levels):

- **Linguistic failure**. This occurs when the NLP component is unable to generate the intermediate representation (but the question can usually be reformulated and answered). This was by far the most common problem, occurring in 27 of the 39 queries not handled by AquaLog (69%).
- **Data model failure.** This occurs when the NL query is simply too complicated for the intermediate representation. Intriguingly this type of failure never occurred, and our intuition is that this was the case not only because the relational model is actually a pretty good way to model queries but also because the ontology-driven nature of the exercise

ensured that people only formulated queries that could in theory (if not in practice) be answered by reasoning about the departmental KB.

- **RSS failure**. This occurs when the relation similarity service of AquaLog is unable to map an intermediate representation to the correct ontology-compliant logical expression. Only 3 of the 39 queries not handled by AquaLog (7.6%) fall into this category.
- **Conceptual failure**. This occurs when the ontology does not cover the query. Only 4 of the 39 queries not handled by AquaLog (10.2%) failed for this reason.
- **Service failure**. Several queries essentially asked for services to be defined over the ontologies. For instance one query asked about "the top researchers", which requires a mechanism for ranking researchers in the lab - people could be ranked according to citation impact, formal status in the department, etc. In the context of the semantic web, we believe that these failures are less to do with shortcomings of the ontology than with the lack of appropriate services, defined over the ontology. Therefore we defined this additional category which accounted for 8 of the 39 failures (20.5%).

4.2 Discussion

Here we briefly discuss the issues raised by the evaluation study and in particular what can be done to improve the performance of the AquaLog system.
Service failures can of course be solved by implementing the appropriate services. Some of these can actually be to some extent ontology-independent, such as "similarity services", which can answer questions like "Is there a project similar to AKT?". Other services can be generically categorized (e.g., "ranking services"), but will have to be defined for specific concepts in an ontology, such as mechanisms to rank people publications, or projects. Here we envisage a solution similar to the one used in the Magpie tool [2], where service developers are given publishing rights to develop and associate services to concepts in an ontology, using semantic web service platforms such as IRS-II [25].
The few RSS failures basically highlighted bugs in AquaLog all of which can be fixed quite easily. A clear example of this is the query "who funds the magpie project", where "who" is understood to be a person, while it of course can also be an organization or funding body.
The few conceptual failures are also easy to fix, they highlighted omissions in the ontology.

The most common and problematic errors are linguistic ones, which occurred for a number of reasons:

In some cases people asked new types of basic queries outside the current linguistic coverage, formed with: "how long", "there are/is", "are/is there any", "how many", etc.

Some problems were due to combinations of basic queries, such as "What are the publications in KMi related to social aspects in collaboration and learning?", which the NLP component of AquaLog cannot currently untangle correctly. Another example is when one of the terms is "hidden" because it is included in the relation but actually it is not part of the relation, as for example in "Who are the partners involved in the AKT project?" One may think the relation is "partners involved in" between persons and projects, however the relation is "involved in" between "partners" and "projects".

Sometimes queries fail because of a combination of different reasons. For instance, "which main areas are corporately funded?", falls within the category of ranking failure, but it is also a linguistic and conceptual failure (the latter because the ontology lacks a funding relationship between research-areas and corporations).

In sum, our estimation is that the implementation of the ranking services plus extending the NLP component to cover the basic queries not yet handled linguistically will address 14 of the 39 failures (35.89%). An implementation of new mechanisms to handle combinations

of basic queries will address another 12 failures (30.7%). Hence, removing redundancies and including also the fixes to the ontology, with both implementations it will be possible to handle 34 of the 39 failures, thus potentially achieving a 87% hit rate for AquaLog. Although a more detailed analysis of these problems may be needed, at this stage it does not seem particularly problematic to add these functionalities to AquaLog.

5 Related Work

We already pointed out that research in natural language interfaces to databases is currently a bit 'dormant' (although see [26] for recent work in the area), therefore it is not surprising that most current work on question answering is somewhat different in nature from AquaLog. Natural language search engines such as AskJeeves [27] and EasyAsk [28] exist, as well as systems such as START [13] and REXTOR [21], whose goal is to extract answers from text. AnswerBus [29] is another open-domain question-answering system based on web information retrieval. FAQ Finder [30] is a natural language question-answering system that uses files of FAQs as its knowledge base; it also uses WordNet to improve its ability to match questions to answers, using two metrics: statistical similarity and semantic similarity.

PRECISE [26] maps questions to the corresponding SQL query, by identifying classes of questions that are easy to understand in a well defined sense: the paper defines a formal notion of semantically tractable questions. Questions are sets of attribute/value pairs and a relation token corresponds to either an attribute token or a value token. Apart from the differences in terminology this is actually similar to the AquaLog model. In PRECISE, like in AquaLog, a lexicon is used to find synonyms. In PRECISE the problem of finding a mapping from the tokenization to the database is reduced to a graph matching problem. The main difference with AquaLog is that in PRECISE all tokens must be distinct, questions with unknown words are not semantically tractable and cannot be handled. In contrast with PRECISE, although AquaLog also uses pattern matching to identify at least one of the terms of the relation, it is still able in many cases to interpret the query, even if the words in the relation are not recognized (i.e., there is no match to any concept or relation in the ontology). The reason for this is that AquaLog is able to reason about the structure of the ontology to make sense of relations which appear to have no match to the KB. Using the example suggested in [26], AquaLog would not necessarily know the term "neighborhood", but it might know that it must look for the value of a relation defined for cities. In many cases this information is all AquaLog needs to interpret the query.

MASQUE/SQL [7] is a portable natural language front-end to SQL databases. The semi-automatic configuration procedure uses a built-in domain editor which helps the user to describe the entity types to which the database refers, using an is-a hierarchy, and then declare the words expected to appear in the NL questions and define their meaning in terms of a logic predicate that is linked to a database table/view. In contrast with MASQUE/SQL AquaLog uses the ontology to describe the entities with no need for an intensive configuration procedure.

6 Conclusion

In this paper we have described the AquaLog query answering system, emphasizing its genesis in the context of semantic web research. Although only initial evaluation results are available, the approach used by AquaLog, which relies on a RSS component able to use informa-

tion about the current ontology, string matching and similarity measures to interpret the intermediate queries generated by the NLP component, appears very promising. Moreover, in contrast with other systems AquaLog requires very little configuration effort. For the future we plan to make the AquaLog linguistic component more robust, primarily on the basis of the feedback received from the evaluation study carried out on the KMi domain. In addition we also intend to carry out a formal analysis of the RSS component to provide a more accurate and formal account of its competence. As already mentioned, more comprehensive evaluation studies will also be needed. Finally, although the ontology-driven approach provides one of the main strength of AquaLog we have also started to investigate the possibility of accessing more than one ontology simultaneously in a transparent way for the user [31].

7 Acknowledgements

The authors would like to thank Maria Vargas-Vera and John Domingue for useful input on AquaLog and related topics. We are also grateful to Kalina Bontcheva for assistance with the use of the GATE NLP component. Finally we would like to thank all those members of the lab who took part in the preliminary evaluation of AquaLog.

8 References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. *Scientific American, 284(5)*, 2001.
2. Dzbor, M., Domingue, J., Motta, E.: Magpie – Towards a Semantic Web Browser. Proceedings of the 2nd International Semantic Web Conference (ISWC2003), *Lecture Notes in Computer Science, 2870/2003*, Springer-Verlag, 2003
3. Mc Guinness, D.: Question Answering on the Semantic Web. *IEEE Intelligent Systems, 19(1)*, 2004.
4. AKT Reference Ontology, http://kmi.open.ac.uk/projects/akt/ref-onto/index.html.
5. Burger, J., Cardie, C., Chaudhri, V., et al.: Tasks and Program Structures to Roadmap Research in Question & Answering (Q&A). *NIST Technical Report, 2001*. PDF available from http://www.ai.mit.edu/people/jimmylin/%0Apapers/Burger00-Roadmap.pdf.
6. Kaplan, J.: Designing a portable natural language database query system. *ACM Transactions on Database Systems 9(1)*, pp. 1-19, 1984.
7. Androutsopoulos, I., Ritchie, G.D., and Thanisch, P.: MASQUE/SQL - An Efficient and Portable Natural Language Query Interface for Relational Databases. In Chung, P.W. Lovegrove, G. and Ali, M. (Eds.), *Proceedings of the 6th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, Edinburgh, U.K., pp. 327-330. Gordon and Breach Publishers, 1993.
8. Chu-Carroll, J., Ferrucci, D., Prager, J., Welty, C.: Hybridization in Question Answering Systems. In Maybury, M. (editor), New Directions in Question Answering, AAAI Press, 2003.
9. Jung, H., Geunbae Lee, G.: Multilingual Question Answering with High Portability on Relational Databases. *IEICE transactions on information and systems, E86-D(2)*, pp306-315, Feb 2003.
10. Androutsopoulos, I., Ritchie, G.D., Thanisch P.: Natural Language Interfaces to Databases - An Introduction. *Natural Language Engineering, 1(1)*, pp. 29-81, Cambridge University Press, 1995.
11. Hunter, A.: Natural language database interfaces. *Knowledge Management, May 2000.*
12. Vargas-Vera, M., Motta, E., Domingue, J.: AQUA: An Ontology-Driven Question Answering System. In Maybury, M. (editor), *New Directions in Question Answering*, AAAI Press (2003).

12.a. Vargas-Vera, M. and Motta, E. (2004). AQUA - Ontology-based Question Answering System. Third International Mexican Conference on Artificial Intelligence (MICAI-2004), *Lecture Notes in Computer Science (LNCS 2972)*, Springer-Verlag, April 26-30, 2004. ISBN 3-540-21459-3.

13. Katz, B., Felshin, S., Yuret, D., Ibrahim, A., Lin, J., Marton, G., McFarland A. J., Temelkuran, B.: Omnibase: Uniform Access to Heterogeneous Data for Question Answering.. *Proceedings of the 7th International Workshop on Applications of Natural Language to Information Systems (NLDB)*, 2002.
14. RDF: http://www.w3.org/RDF/.
15. Mc Guinness, D., van Harmelen, F.: OWL Web Ontology Language Overview. W3C Recommendation 10 (2004) http://www.w3.org/TR/owl-features/.
16. Motta, E.: *Reusable Components for Knowledge Modelling.* IOS Press, Amsterdam, The Netherlands. (1999).
17. WebOnto project: http://plainmoor.open.ac.uk/webonto.
18. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics ACL'02.* Philadelphia, 2002.
19. Tablan, V., Maynard, D., Bontcheva, K.: *GATE --- A Concise User Guide*. University of Sheffield, UK. http://gate.ac.uk/.
20. Katz, B., Lin, J.: Selectively Using Relations to Improve Precision in Question Answering. *Proceedings of the EACL-2003. Workshop on Natural Language Processing for Question Answering*, 2003.
21. Katz, B., Lin, J.: REXTOR: A System for Generating Relations from Natural Language. Proceedings of the ACL 2000 Workshop of Natural Language Processing and Information Retrieval (NLP&IR), 2000.
22. Klein, D. and Manning, C. D.: Fast Exact Inference with a Factored Model for Natural Language Parsing. *Advances in Neural Information Processing Systems 15*, 2002.
23. Fellbaum, C. (editor), *WordNet, An Electronic Lexical Database.* Bradford Books, May 1998.
24. Cohen, W., W., Ravikumar, P., Fienberg, S., E.: A Comparison of String Distance Metrics for Name-Matching Tasks. In *IIWeb Workshop 2003*, PDF available from http://www-2.cs.cmu.edu/~wcohen/postscript/ijcai-ws-2003.pdf, 2003.
25. Motta, E., Domingue, J., Cabral, L., Gaspari, M.: IRS–II: A Framework and Infrastructure for Semantic Web Services, 2nd International Semantic Web Conference (ISWC2003), *Lecture Notes in Computer Science, 2870/2003*, Springer-Verlag, 2003.
26. Popescu, A., M., Etzioni, O., Kautz, H., A.: Towards a theory of natural language interfaces to databases. *Proceedings of the 2003 International Conference on Intelligent User Interfaces*, January 12-15, 2003, pp. 149-157, Miami, FL, USA.
27. AskJeeves: http://www.ask.co.uk.
28. EasyAsk: http://www.easyask.com.
29. Zheng, Z.: The AnswerBus Question Answering System. *Proc. of the Human Language Technology Conference (HLT 2002).* San Diego, CA. March 24-27, 2002.
30. Burke, R., D., Hammond, K., J., Kulyukin, V.:. Question Answering from Frequently-Asked Question Files: Experiences with the FAQ Finder system. *Tech. Rep. TR-97-05, Department of Computer Science, University of Chicago,* 1997.
31. Waldinger, R., Appelt, D. E., et al.: Deductive Question Answering from Multiple Resources. In Maybury, M. (editor), *New Directions in Question Answering*, AAAI Press, 2003.