On the role of a Librarian Agent in Ontology-based Knowledge Management Systems

Nenad Stojanovic
Institute AIFB,
University of Karlsruhe,
76128 Karlsruhe, Germany
nst@aifb.uni-karlsruhe.de

Abstract: In this paper, we present an agent-based approach for the improvement of searching in an ontology-based knowledge management system. The system implements a library scenario in which users query the repository for knowledge resources. Consequently, the so-called Librarian Agent plays the role of the human librarian in the traditional library – it uses all possible information, about the domain vocabulary, the behaviour of previous users and the capacity of the knowledge repository, in order to help users find the resources they are interested in. We partially implemented the approach in the Web Portal of our Institute and some initial evaluation results are shown.

1. Introduction

The basic problem is that the searching for knowledge provided by traditional IR systems only partially reflects the process which humans use in searching for goods in the bricks-and-mortar environment [Jan98]. Briefly, in the non-virtual search, there exist a shop assistant, who helps the user to express his need more clearly and guides the user through the searching space.

Although some recent work is done in the query expansions [Bru97] and recommendation systems [Bal97], the whole process of simulating bricks-and-mortar environment is not modelled so far. The most crucial differences between searching for information in a knowledge management system and the non-virtual searching are:

1) the absence of the support for more precise explicitisation of the user's information needs [Sar75]

2) not targeting the capacity and organisation of the information repository at all.

In this paper, we present an approach for coping with these two issues, in order to enable more efficient searching for knowledge in a knowledge management system. As in the bricks-and-mortar environment, the main module is a (software) agent which analyses the information about the user's activities in the portal and the capacity of the information repository. Based on these analyses, the agent, through an interactive dialogue, guides the users in more efficient searching for information. Particularly, for a

query given by a user, the agent measures its ambiguity, regarding the underlying vocabulary (i.e. ontology), as well as the content (capacity) of the information repository. In case of high ambiguity, the agent suggests the user the most effective reformulation of the query, considering the underlying vocabulary, the information repository and the agents' experience (past behaviour of users). Last, the agent analyses the users' requests off-line and compares the users' interests with the capacity of the information repository, in order to find which "new titles" should be obtained or which topics are no more interesting for users. As in the real scenario, the agent does not perform interviewing of users, i.e. it does not require explicit feedback of users. In order to avoid disturbing users, the agent uses the information stored in the log file of the KM system. The logging system is based on the logging web-server transactions and it is independent of the concrete KM system. The traditional web-server logging approach is extended by capturing the semantic information about pages which the user has visited so far, so that the agent can perform various semantic analyses about the users' behaviour. Since the agent learns from the behaviour of all users, he can make the personalised recommendations, but also the recommendations to the anonymous users. Moreover, the agent can evaluate his own recommendations and consequently improve (learn) his service continually.

The paper is organised as follows: In Section 2, we elaborate on the requirements for more efficient searching in an ontology-based knowledge management system, whereas in Section 3, we discuss the role of a Librarian Agent in fulfilling these requirements. Section 4 contains concluding remarks.

2. The efficient querying in a KM system – the requirements

The problem of satisfying a user's requirement posted to the Information Portal [Bae99] is the question of whether a relevant information resource exists in the information repository, and if the answer is positive, whether the resource can be found by a user. Therefore, the efficient searching depends on:

1. the "quality" of the information repository in the portal,
   - if information resources reflect the needs of users, e.g. if the information repository contains information resources which users are interested in and
2. the "quality" of the searching process, i.e. when a relevant information resource exists in the repository, how easily (if any) the resource can be found. This problem can be divided into two sub-problems:
   a) if a resource which is relevant for the user's information need can be found by the querying mechanism and
   b) if the resource which is highly relevant for the user's information need can be found easily by the user in the list of retrieved results.

The first criterion (1) is the matter of the so-called "collection management policy"[1], which manages the deletion of old information resources and enter of new ones, corresponding to the changes in the user's interests.

[1] For an example, see [SSI02]

The retrieval of resources which are relevant for the user's need (2a) depends on:

1) the clarity of the expression of the need in the query which is posted to the system [Bae99], [Wen01]

2) the quality of the annotation (indexing) information resources in the repository

The part of this problem, a so-called prediction game between providers and users of information, can be resolved by using a commonly-agreed vocabulary, i.e. an ontology [Gua95] as the semantic backbone of the portal. We assume that such an ontology exists in the given domain, and that the system, consequently, benefits from using such a conceptual structure in searching for information [Gua99].

Finding (easily) an information resource which satisfies the users' information need (2b) depends on the capability of the system to interpret the term "relevance" in the right manner.

In the rest of the paper, we are focused on the two issues elaborated in the introduction: the quality of the knowledge repository and the clarity of the users' queries.

3. The Librarian Agent

3.1 The role of the Librarian Agent

As more elaborated in the introduction, the role of the Librarian Agent is:

a) to support the disambiguation of the queries posted by users (query management) and

b) to enable the changes in the knowledge repository regarding the users' information needs (collection management)

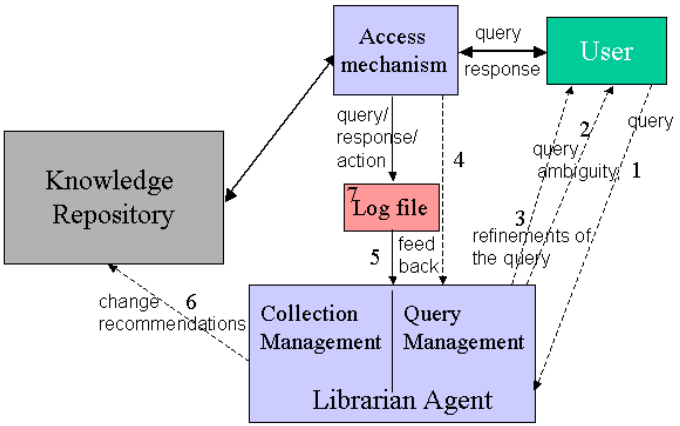

Figure 1. The roles of the Librarian Agent in the process of searching for knowledge

Figure 1 sketched the application scenario for the Librarian Agent. A user posts the query (cf. 1 in the Figure 1) which is first processed by the Librarian Agent. The Agent measures the ambiguity of the query (cf. 2 in the Figure 1) by considering the capacity of the knowledge repository and the domain vocabulary - ontology (for more details, see the next section). In case that the ambiguity of the query exceeds the given threshold, the agent recommends the user some changes (refinements) in the query (cf. 3 in the Figure

1). The Agent receives the feedback information about which suggestions the user has accepted (cf. 4 in the Figure 1), and it uses this information to refine its own strategies for creating recommendations

As the result of the querying, the access mechanism retrieves a list of resources, which is analysed (cf. 5 in the Figure 1) by the Librarian Agent in order to make recommendations for the changes in the collection (cf. 6 in the Figure 1). This recommendation takes into account the analysis of the queries posted by users and the used vocabulary, as well.

In order to avoid disturbing the users by additional questioning, all feedback information is collected implicitly by analysing the activities of the users captured in the log file of the system (cf. 6 in the Figure 1).

In the next section, we give more details about our approaches for the query management.

3.2 The Query Management

3.2.1 Measuring Query Ambiguity

Recent analyses [Cro02] have shown that the precision in searching for information depends strongly on the clarity of the query which a user posts to the system. When the query is formulated in an ambiguous manner, one can expect that a high percent of irrelevant results can be retrieved, independently of the mechanism which is used for searching. Therefore, we see the query disambiguation as the initial step in searching for information in a KM system.

The Librarian Agent observes the query ambiguity in two dimensions:

1) the structure of the query
2) the content of the knowledge repository

1) The structure of the query

Regarding ambiguities in the structure of the query, two other issues are defined:

a) structural ambiguity

   the structure of a user's query is analysed regarding the underlying ontology; in [Sto02a] we define three such criterions: compactness, completeness and aggregation.

b) semantic ambiguity

   the terms form a user's query is analysed regarding the relation which exist in the underlying ontology. In [Sto02b] we have defined two measures for estimating the semantic ambiguity of an ontology-based query: i.e. Clarity and ContextClarity. Very briefly, the Clarity is inversely proportional to the number of subconcepts of a concept and ContextClarity is inversely proportional to the number of relations between two concepts.

2) The content of the knowledge repository

The ambiguity of a query posted in a knowledge repository is obviously repository-dependent. For example, when a user interested in the competitors in 2002 Soccer World Cup gives the query "World Cup" against the collection of the news articles in which the articles about Chess World Cup Tournament are predominant, it is simply impossible for the system to return soccer articles consistently ranked higher than related to chess.

We introduce the Response factor for taking into account the specificities of the knowledge repository content in determining the ambiguity of a query.

The Response factor of a query Q is the measure how the terms from that query cluster the resources in the underlying knowledge repository (KR)

$$\mathrm{Re}\,sponse(Q,KR) = \min_{allQ'} P(not\ \mathrm{Re}\,levant(Q,KR)/(\mathrm{Re}\,levant(Q',KR))) =$$

$$= \max_{allQ'} \frac{NumberOf\ \mathrm{Re}\,levant(Q',KR) - NumberOf\ \mathrm{Re}\,levant(Q,KR)}{NumberOf\ \mathrm{Re}\,levant(Q',KR)}$$

whereas $NumberOf\ \mathrm{Re}\,levant(X,KR)$ denotes the set of knowledge resources stored in the KR which are annotated with the X – in other words, it is the number of results by querying for X the repository KR and $Q' = \{x, x \subset Q, x' \neq \{\}\}$ is the set of all non-empty subsets of the Q. The special case is when the Q contains just one term

$$\mathrm{Re}\,sponse(Q,KR) = P(Q) = \frac{NumberOf\ \mathrm{Re}\,levant(Q,KR))}{TotalNumber(KR)}$$

whereas $TotalNumber(KR)$ is the total number of resources in the repository.

The Response factor describes the probability that a knowledge resource relevant for the query Q will not be relevant for the one of the non-empty subsets of the Q –

$$P(not\ \mathrm{Re}\,levant(Q,KR)/(\mathrm{Re}\,levant(Q',KR))$$

When this probability is very low, it means that the query Q is "covered" by a Q, i.e. that query for Q results in almost all results for the query Q' and that query Q should be extended (refined), in order to return more precise results. The difference between Q' and Q is the set of terms which effect the querying process very low, and probably they should be refined (see the next section). However, this is only a recommendation how to get results which are closer to the information need of the user – it is possible that the user is satisfied with the original query.

3.2.2 Query Refinement

For each query posted to the system, Librarian Agent checks the structural ambiguities, and if they are present, it suggests the improvements of the structure of the query. Next, the Agent calculates ambiguity of the query and gives the users the recommendations how to change the query in order to refine their information needs. Nevertheless, the users can initiate the query refinement on their own.

The Librarian Agent uses three strategies for the query refinement:

A. according to the structure of the underlying ontology

B. according to the content of the knowledge repository

C. according to the users' behaviour (usage - query refinement done by users)

A. The structure of the ontology

In case that the query contains a structural ambiguity, it has to be resolved by considering the structure of the ontology. For example, when the query is incomplete, the Librarian Agent recommends the expansion of the query in such a manner that completeness is achieved. For more details see [Sto02b].

B. The capacity of the knowledge repository

The extension of the query terms should correspond to the characteristics of the document term space. The most popular method in the information retrieval is the so-called local context analysis [Sal90] in which the top-ranked documents are used for the query extension. The extension is usually done by using a variation of the Rocchio coefficients [Roc71]. The query is extended by increasing the influence of the most frequently appearing terms from the top-ranked (i.e. relevant) documents, and by decreasing the influence of the terms from non-relevant documents.

In Section 4.2.1, we have defined the Response factor as the measure of the ambiguity regarding the knowledge repository. Obviously, it is very useful for refining the query.

In case that a query Q is ambiguous because the Response factor is too low, i.e. there is a sub-query Q' for which the list of retrieved results is almost identical to the list of Q, then such a query can be refined by refining the terms for the set containing the difference between Q and Q'.

However, since our approach is ontology-based, the extension we provide is ontology-based, as well. We are not interested in the frequently appearing terms per se, but in the semantic extensions of those terms. For example, when the query is about "ontology+researcher" and the most frequently appearing terms in the top-ranked documents (beside "ontology" and "research") are "professor", "assistant" and "student", which belong to the hierarchy of the concept "researcher", probably the relevant strategy is to expand the query with the information about the "researcher".

C. The users' behaviour

By searching the portal, a user makes a query, observes the list of retrieved results, probably refines the query in some manner, then observes the new list and "clicks" on the information resource when she notes a relevant one; when not, she refines the query again... This is the ordinary user's behaviour, and can be very useful for predicting what can be relevant for a user in a situation. By analysing such information, the system can learn how to rewrite a query in case the user is not satisfied with the retrieved results.

We define three types of query-rewriting patterns based on the users' behaviour described in the rest of the section: expansion-, reduction- and generalization/specialization- pattern.

Considering that a query represents an interest of the user, we can assume that two users who make the same query have the similar interests, regarding the query (situation). They also have the same goal in the searching – to find an information resource about the topic of interest. This assumption allows us to make another one, about the behaviour of users during searching: users with similar interests (goals) should behave in a similar manner. Consequently, for a given query, the system can suggest a user to repeat the

behaviour of the users who have already posted the same query. For example, when a lot of users **expand** the initial query "aspirin and headache" with the term "young" or "old", in order to get more precise results, we can conclude that the treatment of headache by aspirin heavily depends on the age of a patient. Every time the user makes such ambiguity in a query, the system should suggest the user to expand the query with the information about age. It is worth noting that this analysis is performed on the ontological level – "young" and "old" are only two values for the property "age" of the concept "patient". The user should be asked to expand the query not only with the terms "young" or "old", but to select any valid value for "age", e.g. "middle-aged".

The same principle can be used for **reducing** too specific queries, in case no results were retrieved for such a query. The case that users often reduce the query "aspirin, headache, female, young" to the query "aspirin, headache, young" can be interpreted as the irrelevance of the patient's gender for curing the headache by using aspirin. The system should recognise this reduction pattern and recommend such a change every time a user makes this ambiguity in a query. By generalising this pattern on the ontological level, the system can process/treat the previously unseen examples. For example, in case the discovered recommendation is that "queries about side-effects of using aspirin in the patient who suffers from rheumatoid arthritis should not contain information about "age", the query "side-effect, rheumatoid arthritis, aspirin, young, male" can be reduced to "side-effect, rheumatoid arthritis, aspirin, male", although the initial query has not been seen previously.

4. Conclusion

In this paper, we presented an agent-based approach for the improvement of the searching in an ontology-based knowledge management system. The system treats a library scenario in which users query the repository for knowledge resources. Consequently, the so-called Librarian Agent plays the role of the human librarian in the traditional library – it uses all possible information, about the domain vocabulary, the behaviour of previous users and the capacity of the knowledge repository, in order to help users to find the resources they are interested in. Based on various analyses, the agent, through an interactive dialogue, guides the users in more efficient searching for information. Particularly, for a query given by a user, the agent measures its ambiguity, regarding the underlying vocabulary (i.e. ontology), as well as the content (capacity) of the information repository. In case of high ambiguity, the agent suggests the user the most effective reformulation of the query. Last, the agent analyses the users' requests off-line and compares the users' interests with the capacity of the information repository, in order to find which "new titles" should be obtained or which topics are no more interesting for users.

We find that this approach represents a very important step in simulating the brick-and-mortar environment and benefiting from applying the practical results obtained in that area in the searching for information in the virtual world. Moreover, this approach leads to the self-adaptive knowledge portals, which can discover some changes from the user's interactions with the system automatically and evolve their structure correspondingly.

5. References

[Bae99] Baeza-Yates, R., Ribeiro-Neto, B., *Modern Information Retrieval*, Addison-Wesley-Longman Publishing co., 1999.

[Bal97] Marko Balabanovic, Yoav Shoham: Content-Based, Collaborative Recommendation. CACM 40 (3): 66-72 (1997)

[Bru97] Bruza PD and Dennis S. Query Reformulation on the Internet: Empirical Data and the Hyperindex Search Engine. In: Proceedings of RIAO97, Computer-Assisted Information Searching on Internet, Montreal, June 1997

[Cro02] Cronen-Townsend, S. and Croft, W.B., Quantifying Query Ambiguity, in the Proceedings of HLT 2002,pp. 94-98.

[Jan98] Janiszewski, C. "The Influence of Display Characteristics on Visual Exploratory Search Behavior," *Journal of Consumer Research,* 25 (3), 290-301, 1998

[Gua95] Guarino, N. and Giaretta, P. 1995. Ontologies and Knowledge Bases: Towards a Terminological Clarification. In N. Mars (ed.) Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing 1995. IOS Press, Amsterdam: 25-32.

[Gua99] N. Guarino, C. Masolo, and G. Vetere, "OntoSeek: Content-Based Access to the Web", *IEEE Intelligent Systems*, 14(3), pp. 70-80, (May 1999).

[Roc71] Rocchio. J. 1971. *Relevance feedback in information retrieval. The Smart Retrieval system---Experiments in Automatic Document Processing*. G. Salton. Ed. Prentice-Hall Englewood Cliffs. NJ. 313-323.

[Sal90] Salton, G. and Buckley, C. *Improving retrieval performance by relevance feedback.* Journal of the American Society for Information Science. 41(4): 288-297, 1990.

[Sar75] Saracevic, T. (1975). Relevance: A Review of and a framework for the thinking on the notion in information science. Journal of the American Society for Information Science, 26, (6), 321-343

[SSI02] SOSIG Internet Catalogue Social Science Information Gateway Collection Management Policy, http://www.sosig.ac.uk/desire/collect.html

[Sto02a] Stojanovic, N., Stojanovic, L. Usage-oriented Evolution of Ontology-based Knowledge Management Systems, ODBASE 2002,

[Sto02b] Stojanovic, L., Stojanovic, N., Maedche, A. A Framework for Change Discovery in Ontology-based Systems. Second International Workshop on Evolution and Change in Data Management (ECDM 2002), held in conjunction with the 21st International Conference on Conceptual Modelling, ER 2002, Tampere, Finland

[Wen01] Wen, J.-R., Nie, J.-Y. and Zhang, H.-J. Clustering User Queries of a Search Engine. WWW10, May 1-5, 2001, Hong Kong.