Proceedings of I-KNOW '08 and I-MEDIA '08
Graz, Austria, September 3-5, 2008

Information Retrieval Services for Heterogeneous Information Spaces

Julian Bahrs, Benedikt Meuthrath, **Kirstin Peters**
(Business Information Systems and Electronic Government, University of Potsdam, Germany
{jbahrs, bmeuthrath, kpeters}@wi.uni-potsdam.de)

Abstract: Many enterprises loose work time because they lack of global search solutions or their solutions are not able to satisfy the needs in a reasonable time. This results in costs for lost work time as well as increased response time. We present a novel approach to federated search engines that use case based reasoning to rerank results according to the searchers needs and therefore leads to a higher quality of search results and faster information retrieval.

Keywords: enterprise search, information retrieval, case based reasoning, web services, heterogeneous information space
Categories: H.3.1, H.3.3, M.0, M.5, M.7

1 Introduction

In this contribution we present an approach for a self learning search (SLS) system based on case-based reasoning (CBR) to improve result merging and ranking in federated search environments in heterogeneous information spaces. This approach is implemented into a service-based architecture for federated search engines.

First, current limitations and problems in enterprise search are outlined [Chapter 2]. Then we present the architecture of a federated search engine based on search services [Chapter 3]. This system is enhanced with profile and context based search [Chapter 4] as well as feedback mechanisms for learning quality. Finally, we outline a CBR system, to improve future queries [Chapter 5].

2 Enterprise Search

Today, many enterprises cannot access their digital information through a unified search infrastructure [Bahrs, 07]. This leads to lost work time resulting in additional costs and increased response time [Feldmann, 04]. However the powerful and widely accepted search tools used in the internet cannot directly be transferred to enterprises. Enterprise search should cover more than the intranet web pages which are available to all employees. In this contribution enterprise search is understood as search over all text content available in electronic form, including an organization's internal websites, its external websites as well as database records, email or documents [Hawking, 04].

The information space of enterprises differs significantly from the web or classic document collections. It lacks a similar or widely acknowledged structure, such as documents or web pages. In enterprises a collection of elements has to deal with different formats, such as documents, web pages, database records stored in various systems, folders, hard disks and databases, each with unique interfaces. This results in a heterogeneous structure and decentrally stored information space. Second, major parts of a company's information are protected by access rights. The deep web, which stands for content available after forms or even logins, can be understood as an equivalent. Even leading web search engines mostly ignore this content. In contrast it is a necessity in an enterprise to access protected information, as it affects major parts of the overall information. This conflicts the idea of a central index for two reasons: While concepts for central identity management are known, most enterprises are stuck with locally managed access rights, resulting in multiple systems to prove the identity of a searcher. Further, the index would have to replicate the complete access rights in order to hide results not intended for search. This is feasible for a document collection where access rights are granted for complete elements. But it clearly is a difficult task with applications and databases, where access rights are more granular. For example a record set of a customer may be viewed by everyone, while orders are only available to selected users.

In general, the quality of search results increases, if smaller domains are addressed, because content acquisition and indexing as well as query processing can adapt to content specifics. However, this is difficult to achieve with an approach to enterprise search based on the described concept of a search engine. Therefore, enterprise search often leads to a situation of federated search of multiple search engines, where a search engine forwards a search term to a number of underlying search engines and collects result sets. This reflects a conceptual analogy to the information space of enterprise and overcomes the described shortcomings. The new approach is described in [Chapter 3]. The drawback of these systems is the limited ability to merge to an overall relevance ranking [Hawking, 04]. This problem is addressed by the self learning approach presented in the following chapters 4 and 5.

3 Architecture of the Self Learning Search Engine

The SLS prototype is a federated search engine with a modular architecture build upon web services. These web services are provided by different information sources. An information source is itself a search engine for a small domain, that needs to provide at least a uniform resource identifier (URI) [Berners-Lee, 05], a ranking value and an optional description per result.

The architecture of the SLS prototype as shown in [Figure 1] is modeled using the Fundamental Modeling Concepts (FMC). FMC is a modeling technique to support the communication about information processing systems. In [Figure 1] a block diagram is shown to describe the compositional structure of the system in terms of active and passive components. Squares are used to model agents that are active and process information, while rounded figures are used to model storages and small circular figures to model communication channels. Both rounded and small circular figures are passive components to keep or transport information. This structure describes which agent can access which data and communicates with other agents via channels or shared storages.

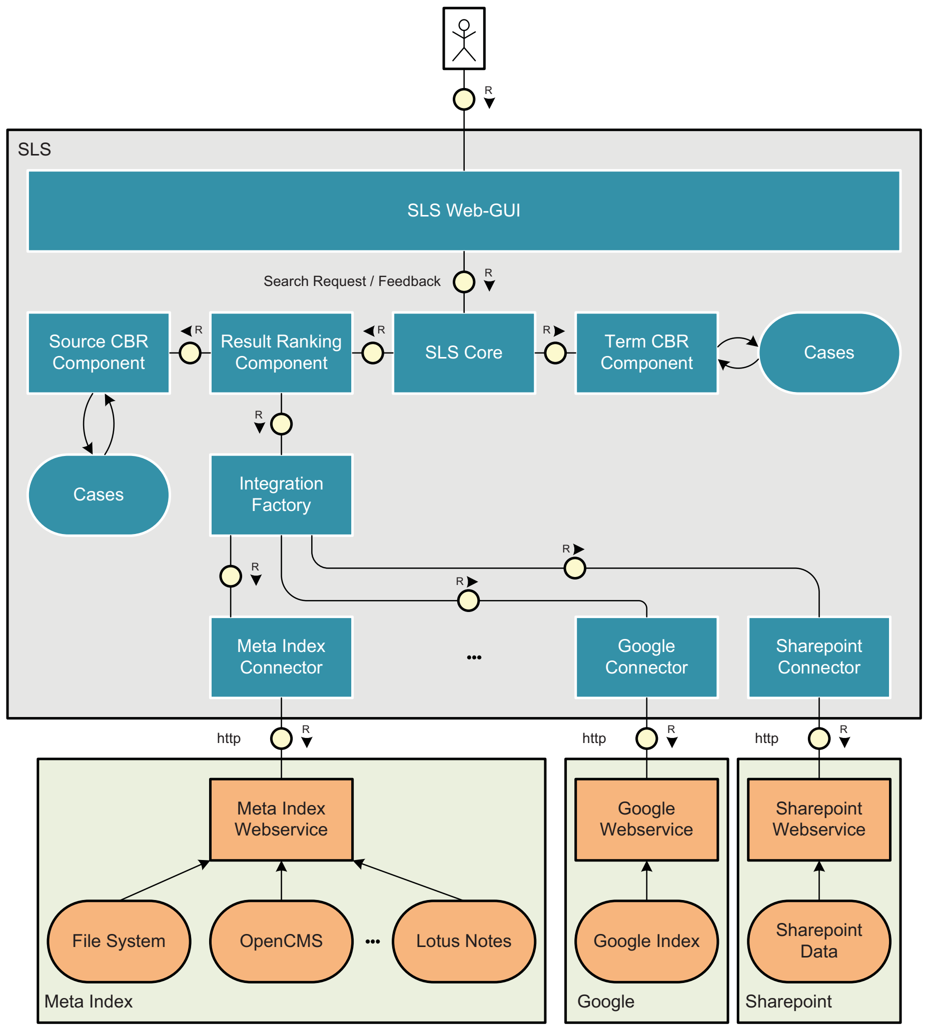

Figure 1: SLS prototype architecture

The SLS prototype consists of a core application, using a web based graphical user interface to communicate with the user. The user is enabled to enter login data that is forwarded to the underlying search engines. Furthermore, for each search request the context has to be chosen [see Chapter 5] and submited together with a search term. The core application forwards this information to a ranking component, which in turn uses the Integration Factory to instantiate connectors for the supported search engine web services. The ranking component collects the search results, merges them into a single result list and reranks the results according to ranking values calculated by the source CBR component. This CBR component uses the z-scores of original ranking value delivered by the web services and retrieved cases to calculate a new ranking [see Chapter 5]. The results are presented as a list to the user. Users are encouraged to give feedback in form of an evaluation of several results. This feedback is forwarded to the source CBR component. On the other side there is a term CBR component, which stores the search terms that lead to positive evaluated results. This is used to suggest similar search terms based on previously learned successful search requests [Gronau, 03].

The connectors can be variable in implementation. The Meta index connector for example connects to a web service that returns results for more than one source, which allows for further slicing and narrowing the search domain in large collections, e.g. SharePoint. As long as the connector returns a single result list with information about the source added to every result, the SLS prototype can handle this.

4 Personalization of search results

The architecture presented above increases the reach of a query and with that also the number of results, because searches are now forwarded to multiple information sources simultaneously. This takes the burden from the user to know and chose the appropriate search engine. However, relevant results may be buried among others. As most users only inspect a very limited number of results, ranking is critical [Joachims, 07]. In our approach ranking is addressed in two places. First, each search service applies its own ranking, which may make use of mechanisms specific to the content of the source. Second, a general reranking is applied based on properties of the user (profile) as well as the situation or goal anticipated with the search query (context).

As with similar approach to personalization, the profile information can be used to rerank or filter a set of results or to enhance the query based on attributes of the user issuing the search query [Riemer, 07]. Most approaches rely on a classification of content and defined profiles of interest. These are either defined prior to search request or are captured by collecting user activity with the search engine [Keenoy, 05]. However, such a profile of interest relies on deeper knowledge of the content and an overall taxonomy, and therefore has to be performed on the level of the search services. To address this personalization (and for handling access rights) we forward information on the users identity to the search services.

On the integrating layer, the selection (preference) of search services can be associated based on profiles. In analogue to the above, these preferences can be either predefined or derived from previous user activity. In our approach we anticipate the later, as it does not require initial set up. We incorporate profiles on the level of a group, instead of individual profiles. This has the advantage that learned improvements can be used for all users with the same profile. We make use of an attribute that is well known within an enterprise: departmental structure. However, searchers may follow different goals with multiple search requests.

We therefore use a second variable that is created or chosen from a list of previously created contexts by the user with each search request. Contexts represent a brief description of the search goals. The list of contexts is maintained per profile and is shared by a group of users.

During set up, an initial set of contexts can be derived through analyzing and labeling all information consuming activities of the members of a profile. One approach to identify and visualize such knowledge activities is the Knowledge Modeling and Description Language (KMDL) [Gronau, 05].

5 CBR System and Learning by feedback

Because of the dynamics in organizations the relevant sources and tasks change overtime, e.g. if a new database is established. The allocation of information sources to tasks needs to be updated permanently. This can be achieved with case-based reasoning.

Case-based reasoning is a machine learning concept and means solving problems by using experiences from similar situations. Roger Schank presented the central role of reminding earlier situations for human reasoning and learning [Schank, 82]. Experiences are saved in form of cases consisting of a problem description, a solution, an evaluation of the solution and any additional information the CBR system may use. The key idea is, that similar problems require similar solutions [Kolodner, 92, Leake, 96, Rissland, 87].

Problem solving with CBR can be divided into four steps illustrated in the CBR cycle [Aamodt, 94] in [Figure 2]: *Retrieve* similar cases from the case base. *Reuse* the information in these cases to solve the new problem. *Revise* the proposed solution and *retain* the new case including problem description, solution and evaluation in the case base for future use [Aamodt, 94].

The CBR system can learn from its experiences, which information sources are relevant for which tasks, by remembering a request for a similar task and the preferred sources in this situation. The task, i.e. the intention of the request, need to be stated by the employee. The profile outlines the general tasks of an employee and with it the general intention of his requests. The context, however, specifies the intention of a request and improves the profile.

To learn which sources are preferred, the CBR system needs some kind of feedback about the quality of presented search results. There are different kinds of feedback discussed in the literature [Kubat, 07, Joachims, 07, Sharma, 05, Kelly, 03]. If the feedback is positive the CBR system will prefer the corresponding source next time.

Our enterprise search engine expects a normalized ranking value for each search result. The CBR system allocates a pushing coefficient to each information source. If the feedback is positive the pushing coefficient of the corresponding source is increased and the results of this source are pushed up in ranking next time, i.e. the normalized ranking values of all results of this source are pushed with this coefficient. If the feedback is negative the pushing coefficient is decreased and the results are pushed down in ranking. [Figure 2] shows the learning cycle of the CBR system.

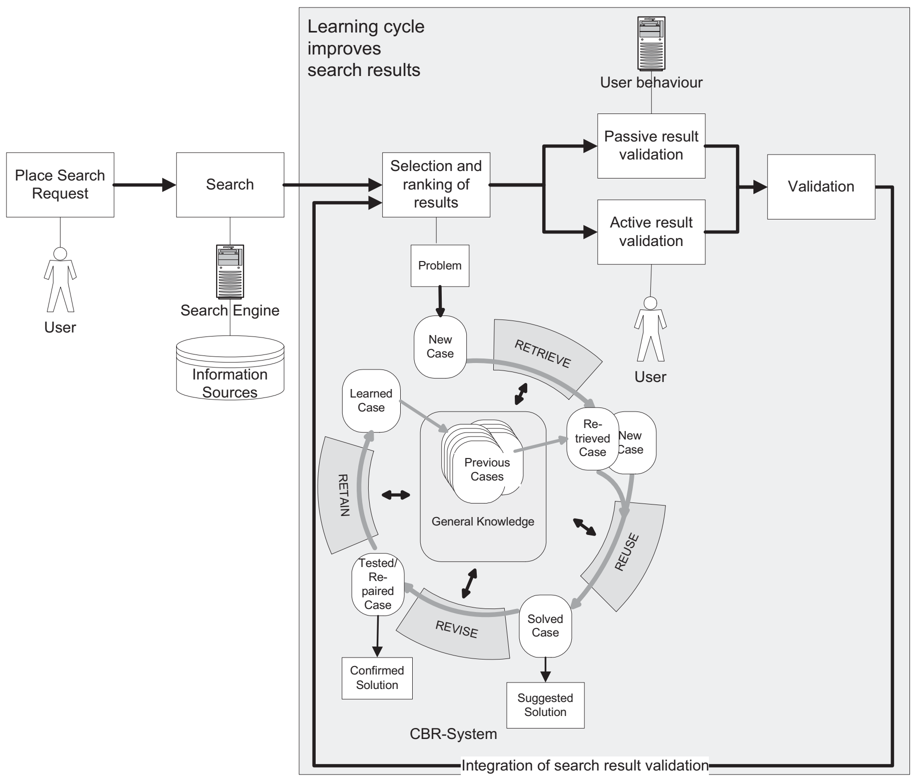

Figure 2: Process of self-learning search engine.

As described above, the intent of a request is used to choose relevant sources. The profile of the searcher and a chosen context is used to characterize this intent, i.e. the problem description of a case consists of a profile and a context. Two cases are equal, if their profile and context are equal. Later we will improve this by additional similarity pointers. To search for a case similar to a new case, profile and context are checked for equality. If no case is found, the new case is added to the case base according to its profile and context.

The solution of a case is a list with a pushing coefficient per information source. The purpose of the coefficients is to push up the information sources in ranking that fits best to the intention of the searcher. Because searchers usually only examine the results with the highest ranking coefficients, pushing the results directly influences the quality of search. Pushing all results of an information source is due to the assumption, that different information sources contain different kinds of information and the ranking within an information source is already done well by the according search service. Obviously the approach needs a high number of information sources to work well. Few sources would lead to few differences between cases.

At the beginning new cases are saved without pushing coefficients. Instead of this an initial coefficient is used, that does not influence the normalized ranking values. The entries of the pushing coefficients are established, as soon as the corresponding information source is evaluated for the first time. This enables an easy enhancement to additional information sources.

Remembering previous cases is also fundamental to learning in CBR. The new case, consisting of a problem description, a proposed solution and an evaluation is saved in the case base.

In addition, another kind of learning is used to enhance the reasoning process. Because the profiles are not completely disjoint, it is possible that two combinations of profile and context describe the same task. It is also possible, that two contexts in one profile describe the same task, because each employee can generate new contexts. To ensure that the list of contexts will not become too long and overlapping, the number of contexts for each profile should be restricted.

The system needs to be trained first in order to build a useful case base. This training phase is completed, if the CBR system reaches a kind a stable state, i.e. the number of cases is almost stable and the coefficients change little and slowly. How long this training phase lasts depends strongly on the current situation. After that the system checks in each retain phase, whether the revised solution of the current case is very similar to another solution in the case base. Two solutions are very similar, if the values of the pushing coefficients are equal for all information sources or differ in only a few values with only little distance. Appropriate thresholds should be generated with respect to the overall number of information sources. Cases with very similar solutions are linked by a similarity pointer and are further treated as equal, i.e. feedback given for one case is also used for every case linked to it. Those cases are used as if they would describe the same situation, i.e. the cases are generalized. Two contexts in one profile, linked by a similarity pointer for a long time, are merged.

The ranking of the results deeply influences, which results are examined [Joachims, 07]. So there is the danger, that only the results of information sources with high pushing coefficients are further examined and evaluated. If we use feedback only to calculate the pushing coefficients, results of sources with bad coefficients are no longer noticed and no feedback is produced for them. Because of the dynamics in organizations, the intention of a task or the content of an information source could change, so it is necessary to reconsider the coefficients, i.e. to relativize not approved pushing coefficients. Pushing coefficients that have not been approved for a long time are reset stepwise to the initial pushing coefficient.

Because of the dynamics in enterprises similarity pointers can become faulty. So they are eliminated, if new feedback contradicts the last feedback of a linked case. Consider e.g. a combination of profile and context treated similar until a new information source is added. If contradicting feedback is stated for this new source, e.g. because the source includes information only relevant for one of the profiles, the similarity pointer linking these two combinations should be removed.

6 Conclusions

Finding relevant information is an actual challenge in enterprises. A federated search engine is an obvious approach to handle the amount of potential information sources. Naturally relevancy depends on the intention of the search engine user. Our approach allows the system to learn the correlation between the search context and the relevance of the information sources. This information leads to a modified ranking of search results, resulting in higher quality result list.

Further research of a mature case base may provide a complete list of all classes of information requirements. This enables the identification of unsatisfied information requirements, e.g. contexts whose information sources were marked irrelevant. This knowledge can be used to improve information landscape. Further key information sources are identified through analyzing the case base.

Our approach is self adapting to heterogeneous information landscapes. It completely relies on self learning algorithms and requires no initial case set up. We circumvent modeling of the context or information need. Instead we assign labels to it and the system learns the preferences. Modeling the information need or assigning information sources would pose a substantial knowledge engineering task, especially if users operate with multiple applications [Fenstermacher, 02].

7 Outlook

A prototypical implementation of SLS on basis of existing open source meta search engines is currently in realization. In a pilot installation the prototype is tested in different stages of development and collects in an extensive log file every single action of all users. The stages of development are as follows: 1. A regular use as meta search engine, 2. with forced validation of the opened links, 3. with personalization and 4. with selected context and reranking by the source CBR component. With this data we want to prove an increase in efficiency in terms of less clicks to a successful hit and a superior validation of hits by the users.

References

[Aamodt, 94] Aamodt, A., Plaza, E.: “Case-Based Reasoning: Foundational Issues”, Methodological Variations, and System Approaches. 1994.

[Bahrs, 07] Bahrs, J., Schmid, S., Müller, C., Fröming, J.: “Knowledge Management in practice – empirical study” (in german). Gito (Berlin), 2007.

[Berners-Lee, 05] Berners-Lee, T., Fielding, R., Masinter, L.: “RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax”. The Internet Society, IETF, January 2005

[Feldmann, 04] Feldmann, S.: “The high cost of not finding information”. http://www.kmworld.com/Articles/ReadArticle.aspx?ArticleID=9534 (Abruf am: 21.11.2007).

[Fenstermacher, 02] Fenstermacher, K. D.: “Process-Aware Knowledge Retrieval”. 35th Annual Hawaii International Conference on System Sciences (HICSS'02), IEEE Computer Society (Island of Hawai, USA), 2002.

[Gronau, 03] Gronau, N., Laskowski, F.: „Using Case-Based Reasoning to Improve Information Retrieval in Knowledge Management Systems“. In: E. Menasalvas, J. Segovia, P. Szczepaniak (Hrsg.): Advances in Web Intelligence. Proc. AWIC 2003, Madrid, May 2003, 94-102

[Gronau, 05] Gronau, N.; Korf, R.; Müller, C.: “KMDL - Capturing, Analysing and Improving Knowledge-Intensive Business Processes”. Journal of Computer Science, 4, 2005; S. 452-472.

[Hawking, 04] Hawking, D.: "Challenges in enterprise search". Klaus-Dieter Schewe, Hugh Williams (eds.): Proceedings Fifteenth Australasian Database Conference, Volume 27. Australian Computer Society, Inc. (Dunedin, New Zealand), 2004, 25-24.

[Joachims, 07] Joachims, T., Radlinski, F.: "Search Engines that Learn from Implicit Feedback". In Computer, 40(8):34–40, 2007.

[Keenoy, 05] Keenoy, K., Levene, M.: "Personalisation of Web Search"; In: Intelligent Techniques for Web Personalization - IJCAI 2003 Workshop, ITWP 2003, Acapulco, Mexico, August 11, 2003, Revised Selected Papers. Springer (Berlin), 2005, 201-228.

[Kelly, 03] Kelly, D. and Teevan, J.: Implicit feedback for inferring user preference: a bibliography. In ACM SIGIR Forum, 37(2):18–28, 2003.

[Knoepfel, 05] Knöpfel, A., Gröne, B., Tabeling, P.: "Fundamental Modeling Concepts. Effective Communication of IT Systems"; John Wiley & Sons Ltd., Chichester (2005).

[Kolodner, 92] Kolodner, J.: "An introduction to case-based reasoning"; In Artificial Intelligence Review, 6(1):3–34, 1992.

[Kubat, 07] Kubat, M., Tapia, M.: "Time spent on a web page is sufficient to infer a user's interest"; In Proceedings of the IASTED European Conference: internet and multimedia systems and applications table of contents, pages 41–46, 2007.

[Leake, 96] Leake, D.: "CBR in Context: The Present and Future. Case-Based Reasoning: Experiences, Lessons, and Future Directions"; pages 3–30, 1996.

[Page, 98] Page, L., Brin, S., Motwani, R., Winograd, T.: "The PageRank Citation Ranking: Bringing Order to the Web". http://dbpubs.stanford.edu:8090/pub/1999-66 (Fetched: 10.12.2007).

[Riemer, 07] Riemer, K., Brüggemann, F.: "Personalization of internet search – techniques and market overview". In: Wirtschaftsinformatik: 49, 2, 2007, S. 116-126.

[Rissland, 87] Rissland, E. and Ashley, K.: "A case-based system for trade secrets law". In Proceedings of the first international conference on Artificial intelligence and law, pages 60–66, 1987.

[Schank, 82] Schank, R.: Dynamic Memory: "A Theory of Reminding and Learning in Computers and People". Cambridge University Press New York, NY, USA, 1982.

[Sharma, 05] Sharma, H., Jansen, B.: "Automated evaluation of search engine performance via implicit user feedback". In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, pages 649–650, 2005.

[Shtykh, 07] Shtykh, R., Jin., Q.: "Enhancing IR with User-Centric Integrated Approach of Interest Change Driven Layered Profiling and User Contributions". In Advanced Information Networking and Applications Workshops, 1, 2007.