Distributed Information Search with Adaptive Meta-Search Engines

Lieming Huang, Ulrich Thiel, Matthias Hemmje, and Erich J. Neuhold

GMD-IPSI, Dolivostr. 15,
D-64293, Darmstadt, Germany
{lhuang, thiel, hemmje, Neuhold}@darmstadt.gmd.de

Abstract. With the flourishing development of E-Commerce and the exponential growth of information produced by businesses, organizations and individuals on the Internet, it becomes more and more difficult for a person to find information efficiently by only using one or a few search engine(s), due to the great diversity among heterogeneous sources. This paper proposes a method for building adaptive meta-search engines with which (1) users can express their information needs sufficiently and easily, and (2) those sources that may answer user queries best can be selected, and the mapping between user queries and the query capabilities of target sources is performed more accurately. Experiments show that this adaptive method for constructing a meta-search engine can achieve more precision than traditional ways. The method can be applied to all kinds of information integration systems on the Internet, as well as on corporate Intranets.

1 Introduction

The revolution of the World Wide Web (WWW or Web for short) has set off the globalization of information access and publishing. Organizations, enterprises, and individuals produce and update data on the web everyday. With the explosive growth of information on the WWW, it becomes more and more difficult for users to accurately find and completely retrieve what they want. Although there are thousands of general-purpose and specific-purpose search engines and search tools, such as AltaVista , Yahoo! , etc., most users still find it hard to retrieve information precisely. Why? Here are some reasons:

1. Users do not know which sources can best answer their information needs. Most users only use a few well-known generic search engines, but each search engine has limited coverage and is not sufficiently capable of coping with information in specific domains.
2. Even though there are some users who collect the URLs (Uniform Resource Locator) of almost all relevant information sources in their bookmarks, still, considering that the user interfaces of these sources differ a lot, it is difficult to expect a user to be familiar enough with the user interfaces and function-

K.R. Dittrich, A. Geppert, M.C. Norrie (Eds.): CAiSE 2001, LNCS 2068, pp. 315–329, 2001.

alities of all sources to use them sufficiently. For this reason, users usually only use a few their favorite sources.

3. The precision of some search engines is too low. A search engine may return thousands or even millions of hits for a user's query. Therefore, picking out the wanted hits is time-consuming and a terrible chore.

How can we overcome such difficulties? An adaptive meta-search engine will effectively and efficiently help all kinds of users find what they want. Such a tool can do several things:

1. It integrates any number of differing information sources (such as search engines, online repositories, etc.). Based on the differences with respect to domain, functionalities, performances, etc., these integrated sources are classified into different groups. When users input a query, the meta-search engine will select those sources that may answer the user query best.
2. It provides users with a uniform user interface to multiple internal and external knowledge sources, thus making the great diversity of various sources transparent to users, improving search efficiency, and reducing the cost of managing information. In order to let users input specific queries and make these queries more close to the target sources, the adaptive mechanism is employed to dynamically construct the user interface.
3. By means of this adaptive mechanism, it can translate user queries into selected sources more accurately. Therefore, users can get more complete and precise results.

In this paper we discuss how we designed such an adaptive meta-search engine. The remainder of this paper is organized as follows. We start by briefly introducing some related work. In section 3 we discuss the architecture of an adaptive meta-search engine and its major components. In section 4, we introduce some experiments carried out for testing the efficiency of a meta-search engine based on this architecture. Finally, section 5 concludes this paper.

2 Related Work

In the Internet there are a lot of meta-search engines such as SavvySearch[1], Dogpile[2], ProFusion[3], Ask Jeeves[4], askOnce[5], to name just a few. Although they integrate a lot of WWW search engines, most of their user interfaces are too simple. They only use a "Least-Common-Denominator" (LCD) user interface, discarding some of the rich functionalities of specific search engines. It is difficult for users to input complicated queries and retrieve specific information. "From the users' perspective the integration

[1] http://www.savvysearch.com
[2] http://www.dogpile.com
[3] http://www.profusion.com
[4] http://www.askjeeves.com
[5] The document company, Xerox. http://www.xerox.com

of different services is complete only if they are usable without losses of functionality compared to the single services and without handling difficulties when switching from one service to another"[6]. In order to avoid losing important functions of search engines, both generality and particularity should be considered when designing a meta-search engine. Some other meta-search engines display the searching controls of all search engines on one page or on several hierarchically organized pages, e.g. All-in-One[7] displays many original query interfaces on a single page. Many efforts have been put into research on information integration, such as [1], [2], [3], [4], [5], [6], [7], [9], etc. However, they do not consider using the adaptive mechanism to construct information integration systems. Compared with previous work, our meta-search engine model has several advantages: (1) the dynamically generated, adaptive user interface will benefit the progressively self-refining construction of users' information needs; (2) conflicts among heterogeneous sources can be coordinated efficiently; (3) user queries will match the queries supported by target sources as much as possible.

3 Constructing an Adaptive Meta-Search Engine

In this section, we first discuss the diversity among Web information sources by investigating the user interfaces of some sources. Then we discuss the architecture of our meta-search engine prototype and its components.

3.1 The Diversity of Sources

At the beginning of this section, six concrete user interface examples are displayed to demonstrate the great diversity among heterogeneous information sources. Fig.1 displays the user interface of the NCSTRL[8] search engine. Fig.2 displays the user interface of the ACM-DL[9]. Although these two search engines are designed for searching computer science papers, we can see many differences between them. ACM-DL provides a more complicated user interface than NCSTRL, so users can input more specific queries by using ACM-DL. In the second CGI form of NCSTRL search engine, each input-box belongs to one specific bibliographic field (i.e., Author, Title, Abstract). While in ACM-DL, there are five check-boxes (each stands for a field, i.e. Title, Full-Text, Abstract, Reviews, and Index Terms.) and users can select one or some of them to limit the scope of the input terms in the input-box. These kinds of differences make query translation from a meta-search engine to a target source difficult.

---

[6] http://www.tu-darmstadt.de/iuk/global-info/sfm-7/

[7] http://www.allonesearch.com

[8] Networked Computer Science Technical Reference Library http://www.ncstrl.org

[9] ACM Digital Library http://www.acm.org/dl/newsearch.html

Fig. 1. NCSTRL

Fig. 2. ACM-digital library

Figures 3 and 4 show the user interfaces of two Internet sources for finding vehicles. Their interfaces are also quite different. Fig. 3 displays a rich-function query interface in which users can set many specific parameters for searching. Fig.4 displays three forms (for browsing, fast searching and advanced searching, respectively). Compared with figures 1 and 2, figures 3 and 4 are more difficult to integrate into a uniform interface because there are more differences between them.

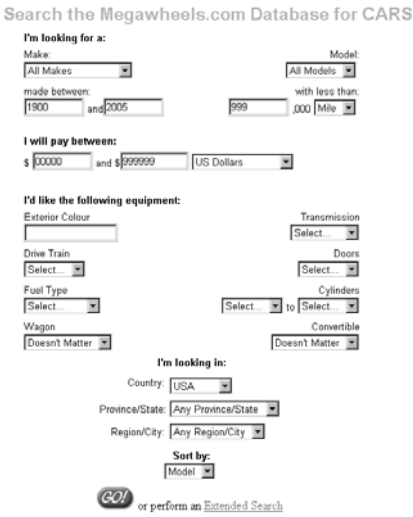

Fig. 3. Megawheels.com

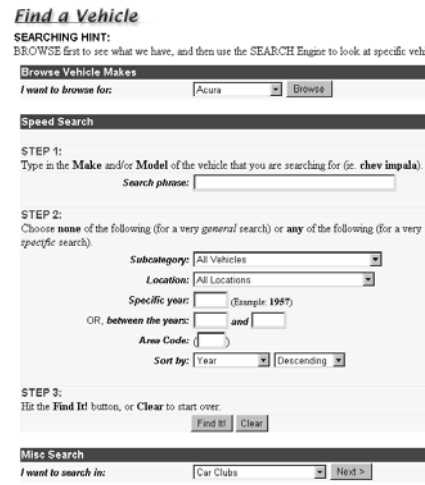

Fig. 4. Carsearch.net

Figures 5 and 6 display two sources for weather forecasting. These kinds of sources do not provide CGI-based query forms, meaning that users can only browse pages for information. However, these semi-structured pages can easily be queried with the

help of wrappers. In section 3.2, we will discuss the wrappers. Most pages on the Web are semi-structured or non-structured, such as product information, personnel information, etc., so that meta-search engine must integrate not only search engines, but also these other kinds of sources.

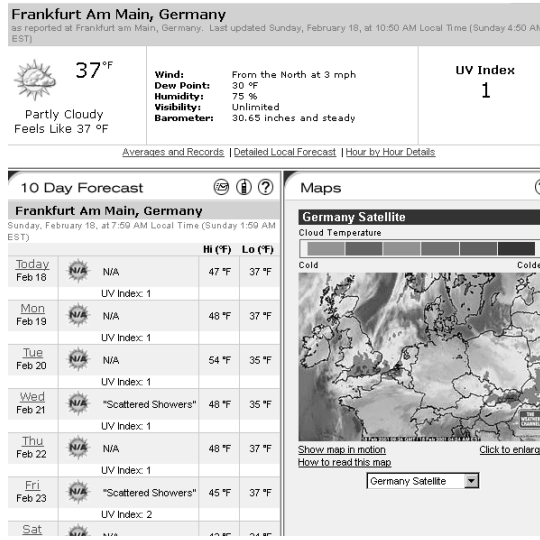

Fig. 5. www.weather.com

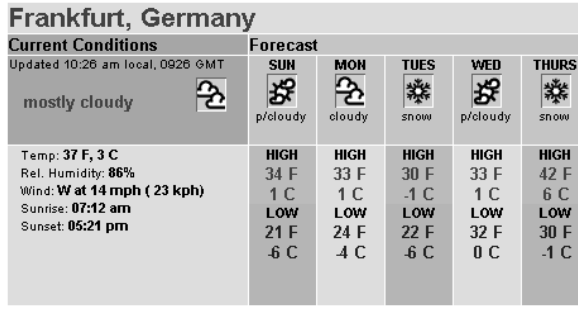

Fig. 6. http://www.cnn.com/WEATHER/

In addition, the diversity of heterogeneous information sources also exists in other aspects:

1. Formats: For example, different search engines return their results using different date formats. Some systems use “September 8, 2000”, others use “09/08/00”, “08/09/00”, “Sep. 2000”, or “08092000”, etc. Some sources use standard names and some use abbreviations (e.g. “kilometer”, “km”).
2. Naming: Different systems use different names for synonyms or homonyms. For examples, some systems use “all fields” to denote this field modifier, some use “anywhere”. In the Dublin Core metadata set, there is only one

element for authors: “Creator”. While in USMARC, there are two elements for authors: “Corporate author” or “Individual author”.

3. Scaling: Different information retrieval systems have different ranking methods. For example, ACM-DL assigns the value 11 to an entry. While the Cora search engine assigns 0.9156 to another entry. How can you compare the relevance of these two entries?
4. Capability: Different retrieval models and query languages. Some sources support Boolean-based queries, some support vector-space-based queries, some support natural language queries. Some sources automatically drop stop-words (e.g. and, with, etc.). Some sources support fuzzy expansion, stemming, right-/left- truncation, or wildcards, and so on.
5. Interface designing: Some sources provide static HTML form user interface, some provide dynamical HTML form user interface, some provide HTML form user interface with JavaScript, and some provide java applet user interface. Some sources provide customizing services for users to personalize their user profiles. Some sources can return all results for a user query, while some sources demand users to visit their web sites more than one time to get complete results. Some sources can let users refine their queries after results come.

From Figures 1-6, we know that there are many discrepancies among the user interfaces of heterogeneous sources and it makes integration difficult. However, all the controls available in user interfaces can be divided by function into three groups:

(1) **Classification Selection Controls**, a classification selection control is a component on the user interface to a search engine, by selecting one or more items of which, users can limit their information needs to certain domains, subjects, categories, etc. For example, in Fig. 2, there is a choice control for users to limit their searches to a certain publication (which proceeding or journal) or all publications. In Fig.3, there are some classification selection controls, such as “Maker”, “Model”, “Country”, “Transmission”, etc.;
(2) **Result Display Controls**, A result display control can be used by users to control the formats, sizes or sorting methods of the query results. For example, in Fig. 1, there is a result sorting control by selecting which the retrieved results can be sorted by “author’s name”, “date” or “relevance”. In Fig. 4, the results can be sorted by “Date of entry”, “Location”, “Make”, “Mileage”, “Model”, “Price”, or “Year”. Some sources provide “Results grouping size” controls;
(3) **Query Input Controls**, All terms, term modifiers and logical operators of a search engine constitute a query input controls group, through which users can express their information needs (queries). A term is the content keyed into an input box on the user interface. A term modifier is used to limit the scope, the quality or the form of a term (e.g. <Title>, <Full-Text>, <Keywords>, <Abstract>, <Author>, <Exactly Like>, <Multiple Words>, <Using Stem Expansion>, etc.). A logical operator is used to logically combine two terms to perform a search, the results of which are then evaluated for relevance. For example, a logical operator can be <AND>, <OR> or <NOT>.

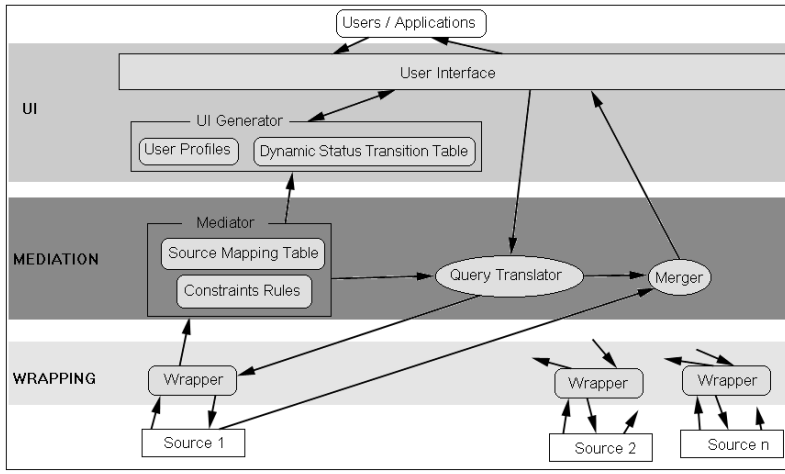

Fig. 7. Architecture of an adaptive meta-search engine

3.2 The Architecture of an Adaptive Meta-Search Engine and Its Components

Fig. 7 displays the architecture of our adaptive meta-search engine. It consists of three layers. The first one is the wrapping layer. Each wrapper describes the characteristics (such as input, output, domain, average response time, etc.) of a source and is responsible for the communication between the meta-search engine and this source. The second one is the mediation layer, which acts as an agent between users and the wrappers. The third one is the UI (user interface) layer that dynamically constructs the query form for users to input information needs. In the following, we will discuss all components in this architecture.

3.2.1 Wrappers – Mediator

The "Mediator [8] - Wrapper" architecture has been used by many information integration systems. The mediator manages the meta-data information on all wrappers and provides users with integrated access to multiple heterogeneous data sources, while each wrapper represents access to a specific data source. Users formulate queries in line with the mediator's global view, that is the combined schemas of all sources. Mediators deliver user queries to some relevant wrappers. Each selected wrapper translates user queries into source specific queries, accesses the data source, and translates the results of the data source into the format that can be understood by the mediator. The mediator then merges all results and displays them to users.

Due to the heterogeneity of Internet information sources, different kinds of wrappers should be employed for different kinds of sources, such as web search engines (e.g. Altavista), web databases (e.g. Lexis Nexis), online repositories (e.g. digital libraries), other meta-search engines (e.g. Ask Jeeves), semi-structured web documents (e.g. product lists), non-structured documents (e.g. Deja UseNet, e-mail installations), and

so on. Each wrapper records the features of an integrated source. Because the information on the WWW constantly changes, this module will periodically check if the user interface of a search engine has been changed, and timely modify the information that describes the query capability and user interface of the search engine.

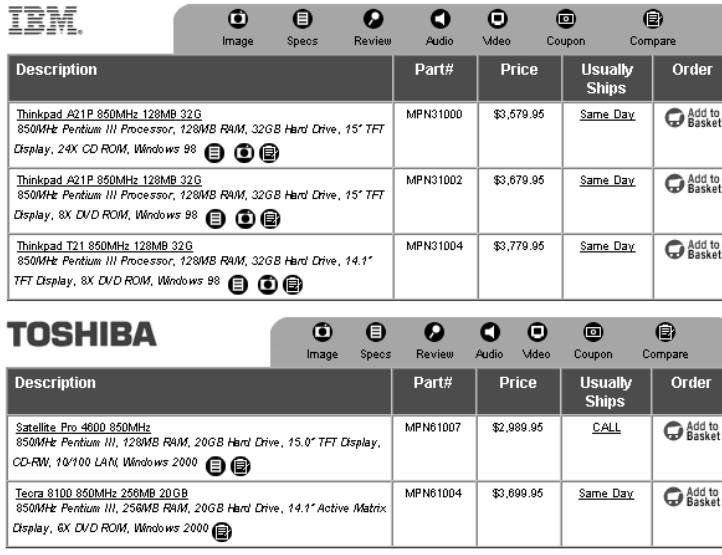

Fig. 8. An example Web page that can be queried by wrappers

In Fig.8, a screenshot of Web page for sailing laptop computers is displayed. From this page, we can extract the information of producers (IBM, TOSHIBA, HP, COMPAQ, etc.), descriptions (CPU, Memory and Storage, Display and Graphics, Multimedia, Communications, etc.), Part#, Prices, etc; and then use wrappers to record such information. The meta-search engine can use an SQL-like query language (e.g. "SELECT * FROM 'http://www.companyURL.com/…' where producer = 'IBM' and price < '$2000' …") to express the information needs of users and extract the relevant information from the pages.

3.2.2 Constraints Rules - Source Mapping Table

Due to the various conflicts existing among heterogeneous sources, a meta-search engine must coordinate these conflicts for the purpose of both constructing a harmonious user interface, and making a more accurate query translation from the meta-search engine to the sources. Constraints rules are employed to record the conflict information between the controls of a source or between different sources. For example, a term belonging to the 'Date' field cannot be modified by the 'Sound like' qualifier, while a term belonging to the 'Author' field can be modified by this qualifier but not by 'Before' or 'After'. When the meta-search engine dynamically constructs the user interface, these constraints rules will be considered in order to eliminate various

kinds of conflicts and to let users input their queries more easily and accurately. Fig. 9 displays three screen shots of the meta-search engine using constraints rules to construct its user interface. In Fig. 9(a), the 'author' field can only be modified by 'Exactly like', 'Sound like' and 'Spelled like' qualifiers. While in Fig. 9(b), the 'date' field can only be modified by 'Before' and 'After' qualifiers and the search terms are two choice controls (one for month and another for year) instead of an input-box.

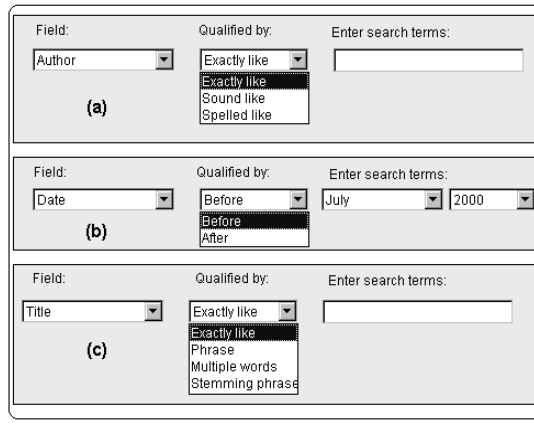

Fig. 9. Three examples of applying constraints rules to UI construction

A source mapping table is used to record the mapping situations between the items of a meta-search engine's choice controls (e.g. category selection controls) and the integrated search engines. For example, if users choose the item <Zoological Journal of the Linnean Society> from a 'Journal' choice control in the meta-search engine, then a car-oriented source will retrieve nothing. When a user has finished the query construction and then submits it, the meta-search engine's mediator will judge which search engines should be used to answer the user's query. If the number of relevant search engines is large, the running priority order will also be decided.

3.2.3 User Profiles – Dynamic Status Transition Table – UI Generator

Because the users of a meta-search engine come from all kinds of application areas, it is favorable for users to be able to personalize their user interfaces. For example, some people have interests only in the field of computer science. In this case, the meta-search engine has to provide users with functionality to customize the query interface, such as selecting relevant search engines and relevant category items of some general-purpose search engines. Users can also set up other parameters like sorting criteria, grouping size, quality of results (layout, file format, field selection, etc). When a user customizes the interface to a certain extent, the resulting interface will be limited to some specific search engines. User profiles are employed to record the configuration that users set.

A dynamic status transition table is used to record the information on user manipulations and user interface status. Depending on the control constraint rules, when a user finishes an action of clicking an item in a control, the system will check the status of all controls. If one condition of a constraint rule can be satisfied, then the items in the right part of the rule are disabled or enabled. Why do we use such a "dynamic status transition table" to manage the information on user interface status and user manipulations? The reason is that various constraints among the controls of a search engine or several search engines need to be coordinated when the user interface of a meta-search engine is being dynamically generated. If there is no such a table, the dynamic interface may be inconsistent with some control constraint rules or even in disorder when the user interface has been changed a lot. This dynamic status transition table can also be used to help users move back to a former status.

Although a static user interface is easy to create and maintain, it lacks flexibility and the interactive nature of an information retrieval dialogue between users and the system. The functionality offered to the users is also limited. During the information retrieval dialogue, the expression of the users' information needs is a self-refining process. Therefore, it is necessary to design a flexible and progressive query interface that is capable of supporting the iterative and self-refining nature of an interactive information retrieval dialogue.

For a progressive query interface, the starting page usually consists of some common controls just like the "Least-Common-Denominator" interface supported by most search engines. During the user query process, the query pages change according to users' needs until the query is finished. Therefore, this kind of query interface has the advantages of both a simple and a sophisticated query interface. Sometimes users only need a simple interface to input keyword(s) without any extra controls. But sometimes users want to input complicated queries and they will complain about the lack of input controls.

The user interface of an adaptive meta-search engine changes in accordance with both the user manipulation and control constraint rules. When users gradually express their information needs by manipulating the controls in the user interface to a meta-search engine, especially some choice controls, the number of search engines that can satisfy the information needs of users may decrease (according to the source mapping table). Suppose that only some of the integrated search engines may be relevant, then the system need not consider the irrelevant controls and items that cannot be supported by these search engines when dynamically constructing the next-step query page. Synthesizing an integrated interface will coordinate the conflicts arising from heterogeneous sources with differing query syntax. There are many differences between the user interfaces and query models of search engines for different domains. For example, it is difficult for a meta-search engine to provide a uniform interface that can be efficiently used by users searching for information on movies, news and architectural engineering. Each time users execute a query, their information needs are on a certain domain or subject. In addition, search engines for similar domains have many similarities in their user interfaces, for example, figures 1 and 2 (they share fields such as 'Author', 'Title', 'Abstract', etc.), figures 3 and 4 (fields such as 'Makers', 'Models', 'Cylinders', etc.), figures 5 and 6 (fields such as 'Temperature',

'Relative humidity', 'Wind', etc.). Therefore, based on the adaptive query model discussed before, an information integration system can facilitate the expression of both the query capabilities of information sources, and the information needs of users. Such an adaptive information system will have higher flexibility and better scalability than traditional ones.

3.2.4 Query Translator – Merger

A user query will be translated into the specific form that a search engine supports. This query translation will consider not only the syntax of the query expression, but also the constraints of term modifiers, logical operators and the order restrictions. This problem needs to be considered more elaborately; otherwise, fully exploiting the functions of heterogeneous search engines will be impossible. Therefore, we should try to make use of the functions of heterogeneous search engines as much as possible. Only thus can we improve the processing speed and achieve more accurate and complete results. This is exactly what meta-search engines should do.

When translating the user query into the format supported by the target source. Sometimes, query subsuming and results post-processing are employed to compensate for the functional discrepancies between a meta-search engine and sources. When the system translates the original query Q^o into the target query Q^t, one of the following three cases will occur:

1. In this case, Q^o can be completely supported by Q^t.
2. Some term modifiers or logical operators in Q^o cannot be supported by Q^t, but after relaxing them (i.e. broadening the scope of the limitation and therefore enabling that more results may be retrieved), the newly-generated Q^o can be supported by Q^t. In this case, the system dispatches the relaxed query, and when the results come, the results are post-processed according to the previous relaxing information. For the relaxed field modifiers, term qualifiers and logical operators, the system uses some filters to record such information and later use them to refine the results in order to compensate for the relaxing of constraints.
3. In this case, Q^o cannot be supported by Q^t even after relaxing some modifiers or logical operators. The system will break Q^o into several sub-queries, then translate and dispatch each sub-query separately. We use some special filters to record such decomposition information. When the corresponding results come, these "special filters" are employed to compose the results.

The "Merger" module translates the results from a remote search engine into the uniform format and performs some post-processing: (1) sorting and grouping all results according to certain criteria; (2) revisiting search engines (some search engines need to be accessed more than one time to get complete information); (3) dynamically reorganizing the displayed results when the results come from some slow-responding search engines; (4) processing the results according to the previous query decomposition; and so on.

4 Experiments

In order to test this applicability of the architecture for meta-search engines, three experiments on different kinds of user interfaces have been carried out. The first experiment (EXPM1) adopted a "Least-Common-Denominator" user interface that contains only a simple input box without constraint controls. (See Fig. 10. There are no modifiers for each keyword, but users can use quotation marks "" to denote phrases.) The second experiment (EXPM2) employed a static HTML user interface (See Fig. 11) containing major controls that may conflict with each other. Almost all current information integration systems employ one of these two kinds of user interfaces. The user interface of the third experiment (EXPM3) was a progressive, dynamically generated Java Applet user interface, in which almost all conflicts are automatically resolved using our approach. (See Fig. 12, which shows four snapshots of user interfaces in which a query has been input progressively. Fig. 12(a) is the initial interface and in Fig. 12(d), the query is completely input.)

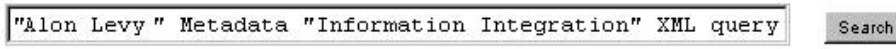

Fig. 10. User interface of the first experiment (EXPM1)

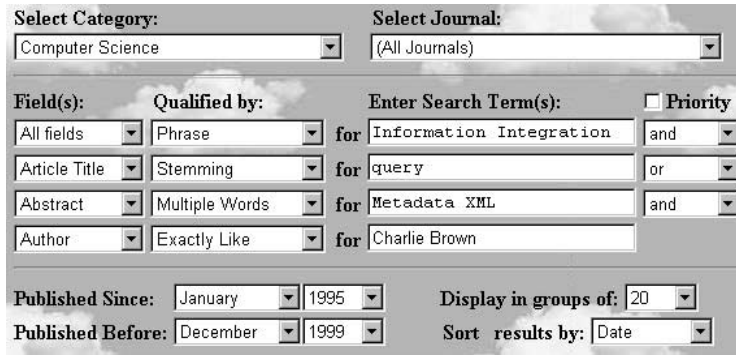

Fig. 11. User interface of the second experiment (EXPM2)

Now we briefly introduce the experimental environments: (1) Seven scientific publication oriented sources were chosen: ACM-DL, CORA, Elsevier, ERCIM, IDEAL, Kluwer, and NCSTRL. (2) Thirty papers were selected from the proceedings of the ACM Digital Libraries, SIGIR, SIGMOD, and VLDB annual conferences between 1995 and 1999. From these 30 papers we chose 45 keywords (including 20 phrases and 25 single words) and 5 authors' names. (3) Thirty-five queries were constructed from these 45 keywords and 5 authors' names according to the actual situation of selected papers. In these queries, most keywords were limited to certain fields. For example, ((**Author is "Charlie Brown"**) *AND* (((**"Information Integration" in All fields**) *AND* (**Title contains "query"**)) *OR* ((**"Metadata", "XML") in Abstract**))) in

the Computer Science category of digital libraries, published during the period of 1995 to 1999, the results to be sorted by date. We judge that a hit is qualified if it is one of the 30 papers or relevant papers.

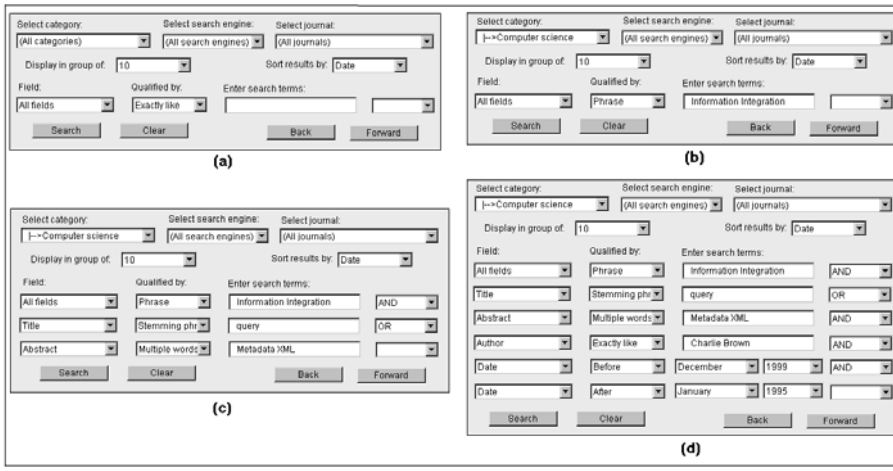

Fig. 12. User Interface of the third experiment (EXPM3)

Table 1. Experimental results

	Returned hits/query	Relevant hits /query	Precision (%)
EXPM1	130.5	1.5	1.1
EXPM2	5.9	1.3	22.0
EXPM3	1.5	1.3	86.7

Table 1 displays the results of the experimental results. The first experiment does not use any kind of constraint information, so it retrieves a lot of irrelevant information. The second experiment can use constraint controls, but it is not flexible enough for users to input queries that are closer to the formats understood by the sources. Because almost all conflicts between different sources or between the controls of the same source have been sufficiently coordinated, the third experiment achieves higher precision than the other two experiments. Table 2 compares the pros and cons of the three kinds of information integration systems with differing user interfaces.

5 Conclusions

Our experiments show that an information integration system with an adaptive, dynamically generated user interface, coordinating the constraints among the heterogeneous sources, will greatly improve the effectiveness of integrated information searching, and will utilize the query capabilities of sources as much as possible. Now,

the adaptive meta-search engine architecture proposed in this paper has been applied to the information integration of scientific publications-oriented search engines. It can also be applied to other generic or specific domains of information integration, such as integrating all kinds of (especially specific-purpose) WWW search engines (or search tools) and online repositories with quite different user interfaces and query models. With the help of source wrapping tools, they can also be used to integrate queryable information sources delivering semi-structured or non-structured data, such as product catalogues, weather reports, software directories, and so on.

Table 2. Comparison of the three kinds of user interfaces for information integration

"Least-Common-Denominator" user interface (Fig. 10)	
Pros	1. It can be supported by all integrated sources; 2. It is simple for users to input information needs and for the system to map queries;
Cons	1. It will inevitably discard the rich functionality provided by specific information sources; 2. It is difficult for users to input complicated queries and retrieve more specific information.
Mixed user interface (Fig. 11)	
Pros	1. Users can express their information needs more accurately than in the "LCD" user interface;
Cons	1. It will increase the users' cognitive load and make the system hard to use for novice users; 2. The constraints between the user interfaces of heterogeneous sources may cause a user query to be inconsistent with a source and make the query mapping difficult; 3. Considering the instability of information sources on the Internet, to maintain it becomes difficult; 4. The static user interface lacks flexibility and makes the interaction between users and system difficult.
Adaptive, dynamically-generated user interface (Fig. 12)	
Pros	1. It has the advantages and avoids the disadvantages of both "LCD" and mixed user interfaces; 2. It will benefit the progressively self-refining construction of users' information needs; 3. Conflicts among heterogeneous sources can be coordinated efficiently; 4. User queries will match the queries supported by target sources as much as possible;
Cons	1. Its implementation is more difficult and time-consuming than the other two.

Such an adaptive meta-search engine can be used in the following spheres:

1. In enterprises, it can help people who are working on market research, decision support and competitive intelligence. By using it, enterprise analysts can simply formulate a single query in a uniform user interface to locate the information they need, rather than accessing several different internal and external sources separately. Therefore, they can easily monitor all marketing and commercial information concerning their businesses worldwide and can answer and respond very quickly to questions on specific subjects.
2. In organizations, researchers, librarians, and other information workers can profit from it.
3. It can serve individuals as a personal web agent.

Acknowledgements

Thanks to Barbara Lutes for discussion on this work.

References

1. Ambite J., Ashish N., Barish G., Knoblock C., Minton S., Modi P., Muslea I., Philpot A., Tejada S.: Ariadne: A system for constructing mediators for Internet sources. In: Proc. of the ACM SIGMOD. Seattle, WA, USA, June 1-4, 1998, pp. 561-563.
2. Adali S., Bufi C., Temtanapat Y.: Integrated Search Engine. Proceedings of the IEEE Knowledge and Data Engineering Exchange Workshop, KDEX97, Newport Beach, CA, USA. November 4, 1997, pp. 140-147.
3. Dreilinger D., Howe A.: Experiences with Selecting Search Engines Using Meta-Search. ACM Transactions on Information Systems. 15 (3), pp 195-222, July 1996.
4. Gravano L., Chang K., Garcia-Molina H., Paepcke A.: STARTS: Stanford Protocol Proposal for Internet Retrieval and Search. In Proceedings of the 1997 ACM SIGMOD, Tucson, AZ, USA. May 11-15, 1997, pp. 207-218.
5. Levy A., Rajaraman A., Ordille J.: Querying Heterogeneneous Information Sources Using Source Descriptions. Proceedings of the 22nd VLDB Conference Bombay, India, pp. 251-262, September 3-6, 1996.
6. Schmitt B., Schmidt A.: METALICA: An Enhanced Meta Search Engine for Literature Catalogs. In Proceedings of the 2nd Asian Digital Library Conference (ADL'99), Taipeh, Taiwan, November 8-9, 1999. pp 142-160.
7. Vassalos V., Papakonstantinou Y.: Describing and Using Query Capabilities of Heterogeneous Sources. In: Proc. 23rd VLDB Conf. Athens, Greece, Aug. 25-29, 1997, pp. 256-265.
8. Wiederhold G.: Mediators in the architecture of future information systems. IEEE Computer, 25(3), March 1992, pp. 38-49.
9. Yerneni R., Li C., Garcia-Molina H., Ullman J.. Computing Capabilities of Mediators. In: Proc. SIGMOD, Philadelphia, PA, USA, May 31-June 3, 1999, pp. 443-454.