Adaptively Constructing the Query Interface for Meta-Search Engines

Lieming Huang, Ulrich Thiel, Matthias Hemmje, Erich J. Neuhold
GMD-IPSI
Dolivostr. 15, Darmstadt, 64293 Germany
{lhuang, thiel, hemmje, neuhold}@darmstadt.gmd.de

ABSTRACT
With the exponential growth of information on the Internet, current information integration systems have become more and more unsuitable for this "Internet age" due to the great diversity among sources. This paper presents a constraint-based query user interface model, which can be applied to the construction of dynamically generated adaptive user interfaces for meta-search engines.

INTRODUCTION
Almost all integrated information systems (IIS) accessing more than one data source employ uniform user interfaces in order to mask the diversity among heterogeneous sources. Most of them provide a "Least-Common-Denominator" (LCD) user interface (See Fig. 1) that can be supported by all sources. The advantage of this method is its simplicity for users when inputting information needs, and for query mappings. However, this method will inevitably discard the rich functionality provided by specific information sources, and it is difficult for users to input complicated queries and retrieve more specific information. This weakness is especially obvious when users want to pinpoint specific information. In order to make full use of the query capabilities of sources and improve the precision of retrieved information, some systems adopt a mixed user interface (See Fig. 4) that integrates almost all controls from various sources. Nevertheless, three obstacles need to be overcome by a mixed user interface: (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, it is hard to maintain mixed interfaces. In addition, the static user interface lacks flexibility and the interactive nature of information retrieval.

Because there is great diversity among heterogeneous search engines, it is difficult or even impossible for a meta-search engine with static uniform user interface to solve such heterogeneity and to utilize the query functionality of all search engines fully. This paper introduces a user interface model **ACQUIRE** (**A**daptive **C**onstraint-based **QU**ery **I**nterface model for integration of heterogeneous sea**R**ch **E**ngines), which sufficiently describes the constraints between the user interfaces and query models of various search engines and can make full use of their functionality by means of the adaptive mechanism. Based on this model, the meta-search engine provides users with a dynamically generated user interface that can adapt itself to the concrete interfaces of relevant search engines during the interaction between users and system. This kind of user interfaces has the advantages and avoids the disadvantages of both "LCD" and mixed user interfaces.

RELATED WORK
Internet meta-search engines, online catalogues, multi-databases and other kinds of information integration systems have attracted a lot of attention since the advent of the network. The issue of providing a common user interface for distributed networked services can trace back to 70s and 80s, such as [3], [5], [6], etc. However, this problem remains unresolved. With the flourishing development of the Internet, many efforts have been put to the meta-searching, such as Savvysearch, AskJeeves, to name just a few. Most of them only use a "LCD" user interface. Some systems only list all user interfaces of different sources separately on a page or several hierarchically organized pages. In [4], Park uses experiments to suggest "it is important to allow for more user controls in various ways in the distributed environment and to characterize different databases to support user choice for integration". In order to avoid losing important functions of search engines, both their generality and particularity should be considered when constructing the user interface of a meta-search engine. Some systems provide more sophisticated user interfaces for information integration, such as [1], [2], etc. However, these systems do not consider the coordination of various constraints among the controls of data sources' user interfaces. Considering the great diversity in schematic, semantic, interface and domain aspects, building an efficient user interface for integration

purposes is quite difficult. It is essential to provide a model that sufficiently describes the query capabilities and user interfaces of heterogeneous data sources. In this paper, the dynamically-generated user interface based on an adaptive query interface model that we propose can achieve the following advantages that the traditional information integration systems do not have: (1) It 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.

ACQUIRE USER INTERFACE MODEL

In this section, we introduce ACQUIRE, a query interface model that we use to describe the query capabilities of heterogeneous search engines.

User Interface Description

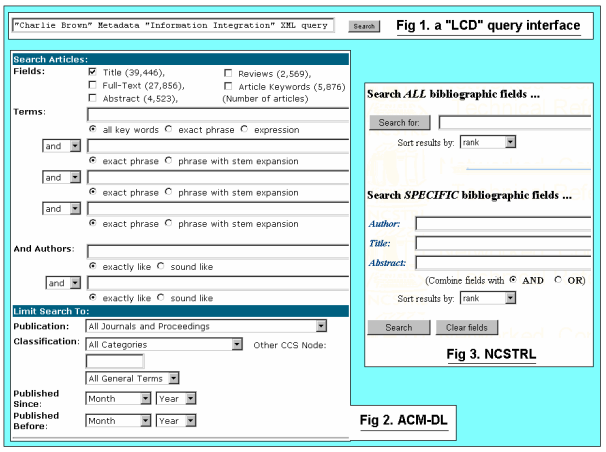

Fig. 1 displays the simple "LCD" user interface that most current web search services employ. Figures 2-3 display the user interfaces of two search engines: ACM-DL (Fig. 2) and NCSTRL (Fig. 3). From these three user interfaces, we can know that it is not an easy thing for a meta-search engine with a uniform interface to utilize the query capabilities of all kinds of search engines to the fullest extent. Although there are a lot of differences between the user interfaces of heterogeneous search engines, we can divide all the controls in the user interface to a source into three groups: (1) classification selection controls, (2) result display controls, and (3) query input controls. In Fig. 4, a user interface for searching scientific publications is displayed to illustrate the following concepts and examples.

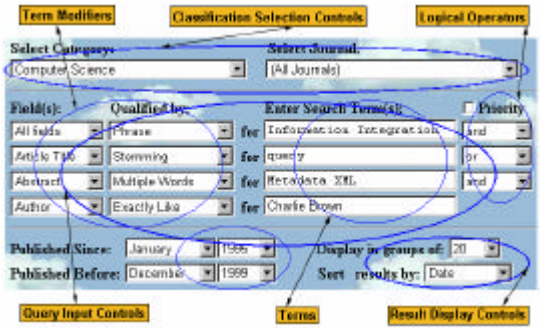

Fig. 4 User Interface of a Search Engine

Classification Selection Controls (CSC)

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, **CSC** = {**Category CSC**, **Journal CSC**, **Search Engine CSC**, **Language CSC**, etc.}.

Result Display Controls (RDC)

A result display control is a component on the user interface to a search engine, which can be used by users to control the formats, sizes or sorting methods of the query results.

For example, **RDC** = {**Sorting Criteria RDC**, **Grouping Size RDC**, **Description RDC**}

Sorting Criteria RDC = {<Relevance ranking>, <Author>, <Date>, etc.};

Grouping Size RDC = {<10>, <20>, etc.}; **Description RDC** = {<full>, <brief>, <URL >, etc.}.

Query Input Controls (QIC)

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). In an adaptive, progressive user interface, the number of these controls may change during users' querying. In the following, we will briefly describe the concepts of "terms", "term modifiers" and "logical operators".

<**Terms (T)**>: A term is the content keyed into an input box on the user interface to a search engine, which is different from the usual meaning of "term" because a term can be a single keyword, multiple words, a phrase, or a Boolean expression. In some cases, the input term may support wildcards, truncation, stemming. It may be case-sensitive, and might drop stop-words, hyphens, diacritics and special characters.

<**Term Modifiers (M)**>: A term modifier is a control on the user interface to a search engine that is used to limit the scope, the quality or the form of a term. For example, (1) **Field Modifiers**: {<Title>, <Full-Text>, < Keywords>, <Abstract>, <Author>, etc.}; (2) **Term Qualifiers**: {<Exactly Like>, <Multiple Words>, <Using Stem Expansion>, etc.}.

<**Logical Operators (L)**>: A logical operator is a control on the user interface to a search engine that 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>.

Control Constraint Rules

There are various constraints among the controls in the user interface of a search engine or user interfa ces of different search engines. Here we only list some cases:

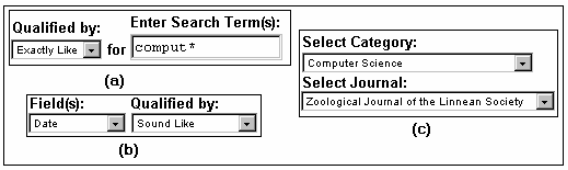

Fig. 5 Invalid modifier for a term

(1) Invalid modifiers for a term. For example, in Fig. 5(a), a term with wildcard cannot use the <Exactly Like> qualifier. If users use the <Exactly Like> qualifier to limit the term "comput*", usually nothing will return.

(2) Incompatible modifiers. For example, in Fig. 5(b), <Date> field cannot be combined with <Sound Like> qualifier. Users can input "Date before 1/1/2000", but what does "Date sounds like 1/1/2000" mean?

(3) Incompatible classification selection controls. For example, in Fig. 5(c), the <Computer Sciences> category with the selection of the <Zoological Journal of the Linnean Society> journal will retrieve nothing.

Suppose there are *n* **controls** in the user interface to a search engine or a meta-search engine, and each control has several **items**. Due to the constraints among the controls of one search engine or among the controls of several different search engines, if users select *p* **items** fromthese n **controls**, there are other *q* **items** from these n **controls** that must be disabled (users cannot select these q items unless they change their previous selection) or must be enabled (these q items are selected automatically). In the following, we list some concrete control constraint rules:

(1)	Field1.**ENABLE**(<Abstract>)→Qualifier1.**DISABLE**(<Sound Like>, <Spelled Like>, <Before>, <After>)
(2)	Category.**ENABLE**(<Computer Sciences>) → Journal.**DISABLE**(<Zoological Journal of the Linnean Society>, <Waste Management&Research>, … …)
(3)	Journal.**ENABLE**(<ACM Transactions on Information Systems>) → Search-Engine.**ENABLE**(<ACM -DL>)
(4)	{Search-Engines.**ENABLE**(<NCSTRL>), Logical-Operator1.**ENABLE** (<AND>) } → LogicalOperator2.**ENABLE** (<AND>)

Architecture of an adaptive constraints-based meta-search engine

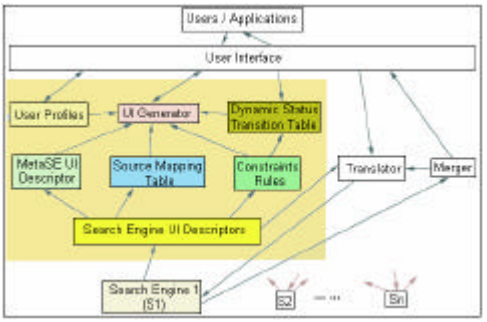

Fig. 6 Architecture of a Meta-search Engine

Fig. 6 illustrates the architecture of a meta-search engine based on the ACQUIRE model. In the following, we briefly introduce some major modules and how they work.

(1) Dynamic Status Transition Table

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. This dynamic status transition table can also be used to help users move back to a former status.

(2) User Profiles

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. 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. For example, students majoring in computer science can customize the new user interface that contains only the computer science category and some relevant journals or repositories. Users can also personalize the layout of the controls. User profiles are employed to record the configuration that users set.

(3) Search Engine UI Descriptors

This module records the features (such as input, output, domain, average response time, etc.) of all integrated search engines. 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.

(4) Source Mapping Table

A source mapping table is used to record the mapping situations between the items of a meta-search engine's classification selection controls (CSCs) and the integrated search engines.

(5) UI Generator -- Adaptive User Interface Generating

In the "Dynamic Status Transition Table" module, we know that the user interface changes in accordance with both the user manipulation and the control constraint rules. When users gradually express their information needs by manipulating the controls (especially the classification selection controls) in the user interface to a meta-search engine, 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 when dynamically constructing the next query page, the system need not consider the irrelevant controls and items that cannot be supported by these search engines. 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. 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. Therefore, taking these characteristics into account, the user interface dynamically generated by such an adaptive meta-search engine can facilitate the expression of both the query capabilities of information sources, and the information needs of users. Moreover, it can have higher flexibility and better scalability than traditional ones.

EXPERIMENTS

The applicability of the proposed architecture was tested by three experiments carried out on different kinds of user interfaces. The first experiment (EXPM1) adopted a "Least-Common-Denominator" user interface that contains only a simple input box without limiting controls. (See Fig. 1) The second experiment (EXPM2) employed a static mixed HTML user interface (See Fig. 4) 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. (Fig. 7 displays four screen shots of the user interfaces in which a query has been input progressively. Fig. 7(a) shows the initial interface and in Fig. 7(d), the query construction is complete.)

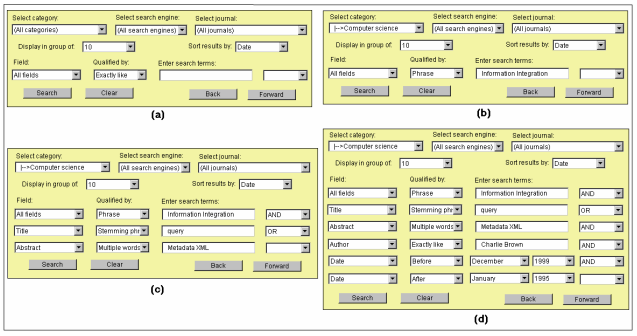

Fig. 7 User Interface of the third experiment

Table 1 Experimental[1] 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 experimental results. The first experiment does not use any kind of limiting controls and the original queries were relaxed a lot, so it retrieves a lot of irrelevant information. The second experiment can use limiting 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. Therefore, the adaptive constraints-coordinated meta-search engine can better support the search for specific information.

CONCLUSIONS

From previous sections, we know that there is great diversity in the user interfaces and query models of search engines for different domains; it is very difficult to unify them. A meta-search engine based on an adaptive constraints-based query interface model will benefit both the expression of users' information needs and the utilization of the information sources' query capabilities to the fullest extent.

REFERENCES

1. Baldonado, M. and Winograd, T. SenseMaker: An Information-Exploration Interface Supporting the Contextual Evolution of a User's Interests, in Proceedings of CHI '97 (Atlanta, GA, April 1997), ACM Press, 11-18.
2. Cousins, S.B., Paepcke, A., Winograd, T., Bier, E.A. and Pier, K. The Digital Library Integrated Task Environment (DLITE), in Proceedings of ACM Digital Libraries (Philadelphia, PA, July 1997), 142-151.
3. Negus, E.A. Development of the Euronet-Diane Common Command Language, in Proceedings of Online Information Meetings (1979), 95-98.
4. Park, S. User Preferences When Searching Individual and Integrated Full-text Databases, in Proceedings of ACM Digital libraries (Berkeley, CA, Aug. 1999), 195-203.

[1] 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. We judge that a hit is qualified if it is one of the 30 papers or relevant papers.

5. Toliver, D. OL'SAM: An intelligent front-end for bibliographic information retrieval. Information. Technology and Libraries 1, 4 (1982).
6. Williams, M. Transparent information systems through gateways, front ends, intermediaries, and interfaces. Journal of the American Society for Information Sciences 37, 4 (1986), 204-214.