Meta Web Search with KOMET

Jacques Calmet and Peter Kullmann

Institut für Algorithmen und Kognitive Systeme (IAKS)
Fakultät für Informatik, Universität Karlsruhe
Am Fasanengarten 5, D-76131 Karlsruhe, Germany
{calmet,kullmann}@ira.uka.de

Abstract. KOMET is a logic-based mediator system that was designed for the knowledge-based integration of heterogeneous information sources [CJKS97]. One of the main challenges in using the numerous information sources in the World Wide Web is to cope with their heterogeneity. In this work we demonstrate how popular web search services like Yahoo and AltaVista can be combined in the framework of KOMET to obtain more useful search results. By gradually increasing the complexity of the meta search application, we demonstrate different features of the KOMET system. Finally, we discuss future developments in KOMET in the light of this example application.

1 Introduction

The concept of a mediator was introduced by Wiederhold [Wie92] and basically denotes a software component that processes information from different information sources in response to a query and combines them in a sensible way. Ideally, this complex process is completely transparent to the user who only directly communicates with the mediator. Usually, wrapper components are used to link information sources into this framework. The wrappers have the task of transforming the mediator query into the source-specific query language on the one hand, and to convert the query results back into the mediator data model on the other hand.

KOMET [CJKS97] is a mediator system which takes a knowledge-based approach to represent and process integration knowledge. This knowledge is expressed in logic programs which are written in the KOMET language. KOMET uses a restricted form of Generalized Annotated Logic [KS92] as logic formalism. Generalized Annotated Logic is a PL1 language and has a high expressiveness due to fact that it can be used with different types of truth values. The only restriction the set of truth values must follow is that it forms a lattice. Syntactically, literals in clauses and facts are explicitely annotated with truth values. The KOMET language uses a clause representation of logic programs and does not support free function symbols. KOMET calculates partial models with regard to a query according to the well-founded semantics using SLG resolution [CW93]. Hence, KOMET is able to deal with arbitrary programs with negation in the rule body and thus allows non-monotonic reasoning. In our framework, information sources are regarded as constraint domains. They can be accessed by using the supplied constraint relations and functions in a clause. The KOMET language serves as versatile means for expressing all the necessary integration steps in a declarative way. We claim that a system like KOMET is ideally suited for establishing complex integrating systems.

In this paper, we show how a meta serch engine for the WWW can be realized in the KOMET framework. We introduce a specialized wrapper component for retrieving web pages and appropriate truth value sets for adequately expressing the integration knowledge. We demonstrate on a sequence of increasingly complex programs how the integration can be achieved.

2 The WWWSEARCH Wrapper

A constraint domain in KOMET is realized by a wrapper component that conforms to the KOMET interface for constraint domains. This interface defines how a relation or function is being called and how results are returned to KOMET.

For our sample application we have realized a wrapper WWWSEARCH which is able to retrieve HTML-pages from a web server and extract pieces of information from these pages. To create a concrete wrapper, WWWSEARCH needs three parameters:

1. The server address.
2. A pattern for the URL on the server for starting a web search.
3. A pattern for extracting URLs from the returned result page.

WWWSEARCH offers only one relation. It is named QUERY and has three arguments: a search term, a URL and the descriptive name of the URL. Since it is not possible and doesn't make sense to start a search without a search term (i.e. placing a variable in the first argument position), we need to prevent the system from doing so. To enforce this kind of restrictions, KOMET allows the definition of binding patterns for relations and functions[1]. For the QUERY relation we define a binding pattern that enforces the first argument always to be bound upon evaluation, whereas the other two arguments must be variables. Evaluation of the relation QUERY in a program will cause the wrapper to establish a HTTP-connection to a server in the WWW and start a query with the specified search term. It will retrieve an HTML-page which was generated by the server containing the first links as result for the query. Using the supplied search pattern WWWSEARCH extracts the URLs and their descriptive names[2] and returns a set of such pairs to KOMET. The WWWSEARCH wrapper is written in C++ and consists of about 200 lines of code.

3 Integrating Information Sources

We distinguish five areas of integration that may be involved in a complex integration task. The following classification scheme reflects our view of information integration which we have found to be useful. To a certain extend it maps to the I^3 reference architecture [HK95]. With its expressiveness, The KOMET language provides the basic means for tackling most of these areas. Where proprietary programming interfaces are involved, a wrapper layer needs to be furnished that makes the functionality available to the KOMET language. To a certain extent, this is supported in our framework by appropriate libraries.

Technical Integration With technical integration we denote the low level mechanisms for accessing information sources, starting queries and retrieving results. This level is a matter of mastering communication protocols and programming interfaces. Usually this level is completely encapsulated by the wrapper components. In our scenario the WWWSEARCH wrapper uses a class library to carry out an HTTP-conversation with remote web servers.

Data Model Integration Once information has been retrieved from an information source, it must be brought into a form which can be processed by the mediator. KOMET uses a common data model into which all information has to be converted. This data model is defined by an interface to which data types have to conform. Typically, data model integration is done inside the wrapper component. In KOMET however, it is possible to define custom data types and thus retain a source specific data format if this is adequate, e.g. if conversions are costly and would impact performance. Any conversions can then be provided by additional functions and formulated explicitely in the mediator program as necessary. The WWWSEARCH wrapper performs data model integration by extracting strings from an HTML page and converting them into instances of the STRING data type of the KOMET data model.

Semantic Integration A challenge of information integration are the semantic or schematic differences information sources might expose. Schema integration is a major issue in information integration research and there exist many approaches to this problem. In our sample however it is of secondary interest.

[1] Binding patterns are sometimes referred to as *modes*. We use these terms synonymously

[2] Normally, the contents of the TITLE tag is used as name for the link.

Conflict Resolution In many situations, information sources may contradict each other. For a sound system it is essential to handle these conflicts to still obtain meaningful results. The problem of merging ranks from different search services falls into this category.

Pragmatic Integration This area comprises postprocessing like aggregation, calculations and analysis to obtain the requested answer. In the meta search application it could mean elimination of duplicates and grouping links that refer to the same server.

In the following section we will develop increasingly complex mediator programs that realize a meta search engine each. We will demonstrate which KOMET features we will exploit to improve our application in each step.

3.1 Representing Information Sources

According to the description of the WWWSEARCH wrapper we define a constraint domain for each search site we want to include. Simple clauses map the query result onto the common predicate SEARCH. Our search engine should allow us to display the source of the link in the result list. We can realize this most easily by introducing a truth value set that consists of combinations of the different information sources. A fact with a truth value $\{Src_1, Src_2\}$ would denote a fact that is true in information source Src_1 and Src_2. This approach has the great advantage that the elimination of duplicates will still work while annotations are fused together automatically. The corresponding program is listed in figure 1.

```
#domains

ALTAVISTA = WWWSEARCH('www.altavista.com','cgi_bin/query?q=%Q',
                      '<dl><dt><b>*. </b><a href="%U"><b>%D</b></a><dd>')
EXCITE = WWWSEARCH('search.excite.com','search.gw?search=%Q','<A HREF="%U">%D</A> ')
YAHOO = WWWSEARCH('ink.yahoo.com','bin/query?p=%Q&hc=0&hs=0','<li><a href="%U">%D</a> -*')
LYCOS = WWWSEARCH('www-english.lycos.com','cgi-bin/pursuit?matchmode=and&cat=lycos&query=%Q',
                  '*<b><a href="%U">%D</A></b>')

#lattices

WWW = POWERSET(AltaVista,Excite,Yahoo,Lycos)

#predicates

SEARCH(STRING,STRING,STRING):[WWW]

#clauses

SEARCH(X,Y,Z):[{AltaVista}] <- ALTAVISTA::QUERY(X,Y,Z)
SEARCH(X,Y,Z):[{Excite}] <- EXCITE::QUERY(X,Y,Z)
SEARCH(X,Y,Z):[{Yahoo}] <- YAHOO::QUERY(X,Y,Z)
SEARCH(X,Y,Z):[{Lycos}] <- LYCOS::QUERY(X,Y,Z)
```

Fig. 1. A Search Engine with Indication of the Source

3.2 Ranking Links

We do not treat the problem of fusing relevance ratings from different sources here in depths. There are various approaches which have been comprehensively discussed in the research community [GGM97]. Eventhough the ratings from different search engines are not easily comparable, we don't want the ranking information to be lost. At least the order of the links could give a hint with respect to its relevance. In our meta search engine we take this rather pragmatic approach. We extend the QUERY relation with another argument in which the position in the result set is returned. Each set of results from a specific search index is then mapped onto the common rating space according to an indvidual parameterization. The parameters for the mapping have been empirically determined. The proposed ranking method is certainly error-prone and should simply be understood as an example of how to implement such a method in KOMET. To represent the rating of a link, we supplement our annotation with a real number from the interval $[0, 1]$. We can construct complex annotations with the parameterized annotation CROSSPR. It allows the definition of an annotation lattice by building the cross product of two or more lattices.

3.3 Grouping Links

It often happens that a number of links are returned that are located on the same web server. Most probably, these links refer to the same subject and it would be convinient to have them grouped together and ideally represented by only one link. To facilitate this, we introduce a new data type, called LINK. It represents an URL together with a link label, the desriptive name of the URL. Introducing a new data type is useful if we need to include additional functionality. As for constraint domains, it is possible to define functions and relations that are logically tied to a specific data type. We change the relation QUERY accordingly, so that it returns LINKs instead of STRINGs. We implement a function SERVER that returns the server name as a string from a given link. For displaying results, a query with the predicate SEARCH_S is issued. For each server in the result set, a query with predicate SEARCH_S_L is started which returns the appropriate list of links. Note, that due to internal caching in KOMET the actual information sources are only queried once in this process [CK99]. The program is illustrated in figure 2.

```
#domains

... (unchanged)

#lattices

WWW = POWERSET(AltaVista,Excite,Yahoo,Lycos)
RANKWWW = CROSPR(REAL01,WWW)

#predicates

SEARCH(STRING,LINK):[RANKWWW]
; same as SEARCH but returns the server name in the second argument
SEARCH_S_L(STRING,STRING,LINK):[RANKWWW]
; returns only the server names for a search term
SEARCH_S(STRING,STRING):[RANKWWW]

#clauses

SEARCH(X,Y):[ALTAVISTA::MAP(R),{AltaVista}] <- ALTAVISTA::QUERY(X,Y,R)
SEARCH(X,Y):[EXCITE::MAP(R),{Excite}] <- EXCITE::QUERY(X,Y,R)
SEARCH(X,Y):[YAHOO::MAP(R),{Yahoo}] <- YAHOO::QUERY(X,Y,R)
SEARCH(X,Y):[LYCOS::MAP(R),{Lycos}] <- LYCOS::QUERY(X,Y,R)

SEARCH_S_L(X,LINK::SERVER(Y),Y):[V] <- SEARCH(X,Y):[V]

SEARCH_S(X,Y):[V] <- SEARCH_S_L(X,Y,Z):[V]
```

Fig. 2. A Search Engine with Ranking and Grouping

3.4 Incorporating More Knowledge

The previous section have mainly dealt with the postprocessing of link list returned by the search engines. Another aspect of integration is the processing of the query before it is send to the indivual information sources. Such a preprocessing could be sensible in the case of a meta search engine if we are interested in pages written in different languages. In this case, the search term needs to be translated accordingly before the actual search is started. Another point of interest could be the inclusion of ontological knowledge to control the search. Using an ontology, the meta searcher could narrow or broaden the search by manipulating the search term if adequate. Additionally an ontology could be used to exploit knowledge about which web index is to be preferred for a certain subject.

For our example we incorporate an English-German online dictionary for translating search terms into the other language and retrieving links for pages in both languages. We create a new domain LEO which queries the dictionary with the relation QUERY and returns translations for a specified search term. The listing is given in figure 3.

```
#domains
... (unchanged)
LEO = DICT('www.leo.org','cgi-bin/dict-search?search=%Q&header=on&links=hide&mirrors=on',
          '<TD VALIGN="TOP">%E</TD><TD VALIGN="TOP">%G</TD>')
#lattices
... (unchanged)
#predicates
... (unchanged)
#clauses
... (unchanged)
SEARCH_S_L(X,LINK::SERVER(Y),Y):[V] <- LEO::QUERY(X,Z) & SEARCH(Z,Y):[V]
```

Fig. 3. A Translating Meta Search Engine

4 Conclusions

We have demonstrated how the KOMET system can be sucessfully used to build a WWW meta search engine that combines the query results from different Internet search services in a sensible way and presents them to the user. This is a typical problem of information integration. Due to the high expressiveness of the KOMET language and the rich features of the KOMET framework this can be achieved with relatively little effort. The modular concept of KOMET facilitates the establishment of a library of components, like domains and data types, that can be reused. Any of the above programs could be easily extended, if a new service would appear in the WWW that is to be included in the application.

One shortcoming of the current KOMET system is that it does not take advantage of implicit potential for concurrent execution of subtasks. Clearly the meta search application would highly profit if the different search indexes would be queried in parallel. However, concurrency is no principle problem in KOMET and will be tackled among other optimization issues in the future. The different meta search engine described in this paper and other information about KOMET can be accessed on our web page `http://calmet-pc.ira.uka.de/komet/`

References

[CJKS97] J. Calmet, S. Jekutsch, P. Kullmann, and J. Schü. KOMET – A System for the Integration of Heterogeneous Information Sources. In *10th International Symposium on Methodologies for Intelligent Systems (ISMIS)*, 1997.

[CK99] J. Calmet and P. Kullmann. A Data Structure for Subsumption-Based Tabling in Top-Down Resolution Engines for Data-Intensive Logic Applications. In *11th International Symposium on Methodologies for Intelligent Systems (ISMIS)*, 1999. Accepted for publication.

[CW93] W. Chen and D. S. Warren. Query Evaluation Under the Well-founded Semantics. In *ACM Symposium on Principles of Database Systems*. ACM Press, 1993.

[GGM97] L. Gravano and H. Garcia-Molina. Merging Ranks from Heterogeneous Internet Sources. In *Proceedings of the 23rd International Conference on Very Large Databases (VLDB)*, 1997.

[HK95] R. Hull and R. King. Reference Architecture for the Intelligent Integration of Information. In *Technical Report, I3 Project*, 1995.

[KS92] M. Kifer and V. S. Subrahmanian. Theory of Generalized Annotated Logic Programming. *Journal of Logic Programming*, 12(1):335–367, 1992.

[Wie92] G. Wiederhold. Mediators in the architecture of future information systems. *IEEE Computer*, 25(3):38–49, March 1992.