User-friendly Information Retrieval in Data Bases and in the World Wide Web [1]

Hermann Helbig
International Computer Science Institute, Berkeley CA
hermann@ICSI.Berkeley.edu *and* hermann.helbig@fernuni-hagen.de
and
Carsten Gnoerlich, Dirk Menke
University Hagen
Applied Computer Science/Artificial Intelligence
D-58084 Hagen
{Carsten.Gnoerlich|Dirk.Menke}@fernuni-hagen.de

March 7, 1997

[1]This is a version of a paper which has been accepted for the AAAI Spring Symposium on Natural Language Processing for the World Wide Web

Abstract

The paper describes two methods for realizing a user-friendly access to distributed information ressources. The first method (Method I) is based on a form-driven dialogue, which is used in the project named "MEDOC". It aims at an experienced user who is familiar with attribute value structures of the data base schemes of typical information retrieval systems (IRS) and who knows the definition of boolean operators. The second method (Method II) applied in the system LINAS is from the very beginning oriented towards natural language communication between end-user and IRS. Both methods can be used in an interface between the user and an information brokering system helping him/her to find an appropriate information provider for his/her demands in networked information systems. Method I gives the user a certain guidance in formulating his/her queries but has a restricted expressive power. It almost never supports the user in automatically finding more complicated descriptional elements, as for instance classificators of a standardized classificational system. Method II, on the other hand, is devoted to the "naive user" having no experience with information retrieval techniques. Allowing for an unrestricted natural language input, it is distinguished by a greater expressive power and gives valuable support in automatically finding descriptors and classificational categories used in the description of documents. In comparison with Method I, there is less guidance in formulating the users demands.

1 Introduction

In general, the design of a database retrieval interface should follow the same design goals as any other modern, interactive software tool. The system should be easy to learn and should communicate with the user in a comprehensible way. On-line help facilities should be invokable from any given state of the system to avoid that an unexperienced user gets stuck at a certain stage of the retrieval process. Consequently, the system should be prepared for the user changing his mind after going half-way through a particular operation, and therefore provide some means of exiting from any incomplete action in a well-defined way.

Another point to take into consideration is that a user's preference with regard to the interface may change depending on his skills already acquired with the retrieval system. Just like a good text editor supplies a menu system for novice users as well as an alternative system of keyboard sequences for the more experienced, the retrieval system should not only concentrate on maximum userfriendliness for beginners, but also provide a user configurable expert mode for professional users.

In contrast to a typical 'stand alone' application like a word processor, it is also remarkable that in general not all components of the retrieval system run locally on the user's machine. Typically, only the interface runs on the user's own machine, while the database is hosted on a dedicated database server connected through a network. Ideally, the implementation details of the underlying database and network system are totally hidden from the end-user by creating a uniform "look and feel" at the user interface level. Therefore, the transformation from the user's query to the target system's representation has to be a process completely behind-the-scenes.

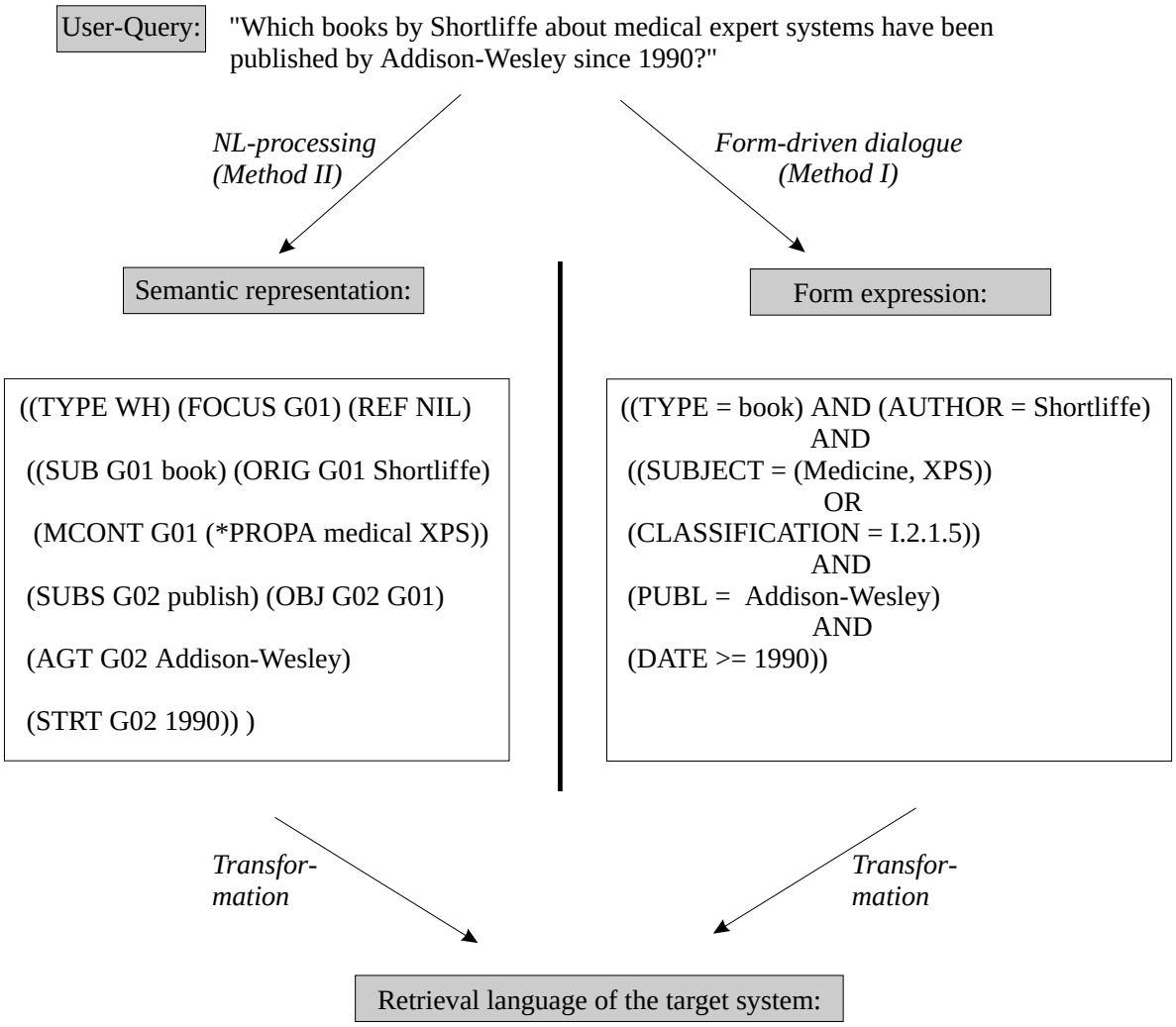
However, presenting a uniform user interface does not mean that also in a technical sense there will be only one specific form of the user interface — it is very likely that users wish to connect to the database using a wide variety of hard- and software components like X11- and Windows-based platforms. Even non-graphical terminals have to be put into consideration - a public library might opt for VT100 terminals as a cost-effective solution, while a natural language retrieval system could even be queried through a normal telephone line using acoustic language recognition for input and speech synthesis for output, respectively. Again, in an ideal scenario, the user would be able to transfer his skills acquired with a specific incarnation of the user interface family to another implementation without having to re-learn the whole system.
In the following we shall concentrate on the needs of an end-user of an information retrieval system (IRS) with regard to the communicative properties of the system.

2 User requirements and two methods of their fulfilment

The long-term aim in the development of an IRS consists in warranting a user-friendly access to the system not only for specialists but rather for the so-called "naive users" having no special computer education. Especially in the work with world-wide distributed IRS, it should be hidden from the user which information source finally answers his/her informational demands. There are two principal approaches in realizing this aim which shall be explained and compared in this paper.
On the one hand it is possible to equip traditional access systems (especially already existing network browsers) with more and more comfortable and intelligent user interfaces (Method I). On the other hand we can place the natural language as a communicational means at the disposal of the user (Method II). Both approaches have to meet the following requirements:

- Simplicity of operation (no special training, ergonomic design of the user interface)
- Realization of a largely natural dialogue (e.g. possibility of paraphrasing)
- Warranting of an adequate expressiveness of the query language (logical properties, permissibility of vague terms)
- Allowance of references (from question to question, from question to foregoing answers)
- Multilinguality (independence from the language of the information source)
- Robustness and cooperativity (error correction, acoustic similarity; observing the correspondence between query type and answer)
- Screening off the technical details (independence of the user from the knowledge about the system).

The first way to meet these requirements (Method I) is followed in the System MEDOC [4]. In this case the user is guided by a form-driven dialogue where the initiative stays with the system.
The user will be provided with a search mask showing a fixed set of predefined attributes and a choice of logical operators to connect the attribute-value combinations entered by the user. The advantage of this approach consists in the explicit guidance of the user through the system while on the other hand the predefined masks impose a certain inflexibility upon the user. Each time he/she wants to switch to another domain of discourse the form shown and the attribute set have to be changed. Within informationally similar domains of discourse the system can be equipped with a standardized set of attributes (compare [5]) but an entirely free dialogue using this method is not realizable. Another problem arises from the need of establishing references between a question and foregoing questions or answers respectively. This possibility is rather restricted in using a form-driven dialogue. By assigning unique indices to the answers of the system it is possible at least to establish references from questions to former answers (this technique, for instance, is applied in the IRS AIDOS [1]). To guarantee a certain multilinguality with Method I, one has to embed elements of linguistic data processing and multilingual thesauri into this surrounding which are scarcely available for most domains of discourse and which, after all, don't fit very well into this method.
The second way to fulfil the above mentioned requirements (Method II) is oriented from the very beginning towards the natural language as the only means of communication a priori well-known to all users (this fact is especially important for "naive users"). Method II is incorporated in the system LINAS [3]. Here, the advantage of universality and great expressiveness must be contrasted with the comparatively huge costs for the preparation of the necessary background knowledge (building of large computer lexicons and comprehensive grammars suitable for automatic natural language processing). From the strategic point of view this is the most promising way to open a user-friendly access to the heterogeneous scene of world-wide distributed information sources for naive users, especially if we think of the acoustic channel of communication which is immanently connected with natural language (Method I is excluded per se if we want to realize an access to the Internet via telephone). With Method II we have to employ the full range of NL processing techniques starting with morpho-syntactic analysis of the user query, including its semantic interpretation and ending with a semantic representation of the question (cf. left side of fig. 1). Because of this algorithmic effort, the dialogical properties of the natural language interface (e.g. allowance of references and elliptical expressions) and the possibility of using paraphrases result almost automatically.
In comparison with that, the costs connected with Method I are not so high, because the input of the user can be more rigidly guided by the form interface according to the syntactic patterns of the formal retrieval language of typical target systems (cf. chapt. 5 and right side of fig. 1). While the natural language approach without doubt has the greater expressivity, the form-driven dialogue possibly shows the better adequacy with regard to the correspondence between the expressiveness of the query language offered to the user and the retrieval language of typical target systems (cf. lower part of fig. 1). Allowing for a natural language communication, it is sometimes difficult to



SELECT { ALL / TITLE, YEAR } FROM LITERATURE WHERE

((DOCTYPE = BOOK) AND

(AUTHOR = "SHORTLIFFE") AND

((DESCR = (MEDICINE; XPS)) OR (CLASS = I.2.1.5)) AND

(PUBL = ADDISON*WESLEY) AND

(YEAR >= 1990))

Figure 1: Comparison of different access methods

prevent the user from asking meta-questions like *What information do you have at all?* or *Why can't you find anything?*. On the other side, it is not so easy with intelligent broker systems having a certain amount of meta-knowledge to include special means of expressivity into the form corresponding just to this meta-knowledge (see chapt. 3). In the following we shall assume as a general scenario that an information mediating system (IMS) stands between user and information provider screening the user from the information source which satisfies the user's demands.

3 The role of background knowledge

To translate the user queries into expressions understandable by whatever information provider reachable through a given network, the IMS has to have some background knowledge or metaknowledge about the information resources:

1. Dialogue situation/dialogue model. At first, the IMS must have some general understanding of the interrelations between the main participants of the information exchange: the user, the IMS itself, the provider and their respective instances as well as the objects exchanged by them (information, money in case of paid services etc.). The principal structure of such a dialogue model using the representational means of semantic networks is shown in fig. 2. An IMS possessing a natural language interface (NLI) must also have some special knowledge about the meaning of deictic pronouns (I - the user, you - the IMS, he - the provider) and of special verbs like "show", "know" etc. establishing the rhetoric context in a natural language dialogue. This knowledge indicated in the upper part of fig. 2 helps to decide which constituents of a NL-query can be dropped ("*What do you know about ...?*", "*Please, show me ... !*") and which parts constitute the informational content really asked for. Also information about special providers belongs to the dialogue model ("*What type of information can they deliver?*", "*What information domain do they cover?*"). The latter part of the model is only hinted at in fig. 2.

2. Metaknowledge about the content of the information sources. The most important knowledge part needed for the transformation of the user queries into expressions of the retrieval language of the provider DBMS is the data base schema defining the structure of the stored information (cf. right side in fig. 3) and the syntax of the retrieval language itself. Together with taxonomic knowledge about attributes and values (lower right side of fig. 3) it is the main source for the transformation of the user query (see chapt. 5).

3. Thesaurus and classification. There are two components of the stock of background knowledge which can be characterized as conceptual ordering systems implicitly or explicitly used in many IRS (left side in fig. 3). The first one, the so called thesaurus, contains the system of concepts describing the information domain (or the field of discourse). It is an alphabetically ordererd set of concepts which are interrelated by certain lexical relations (e.g. by SUB - conceptual subordination,

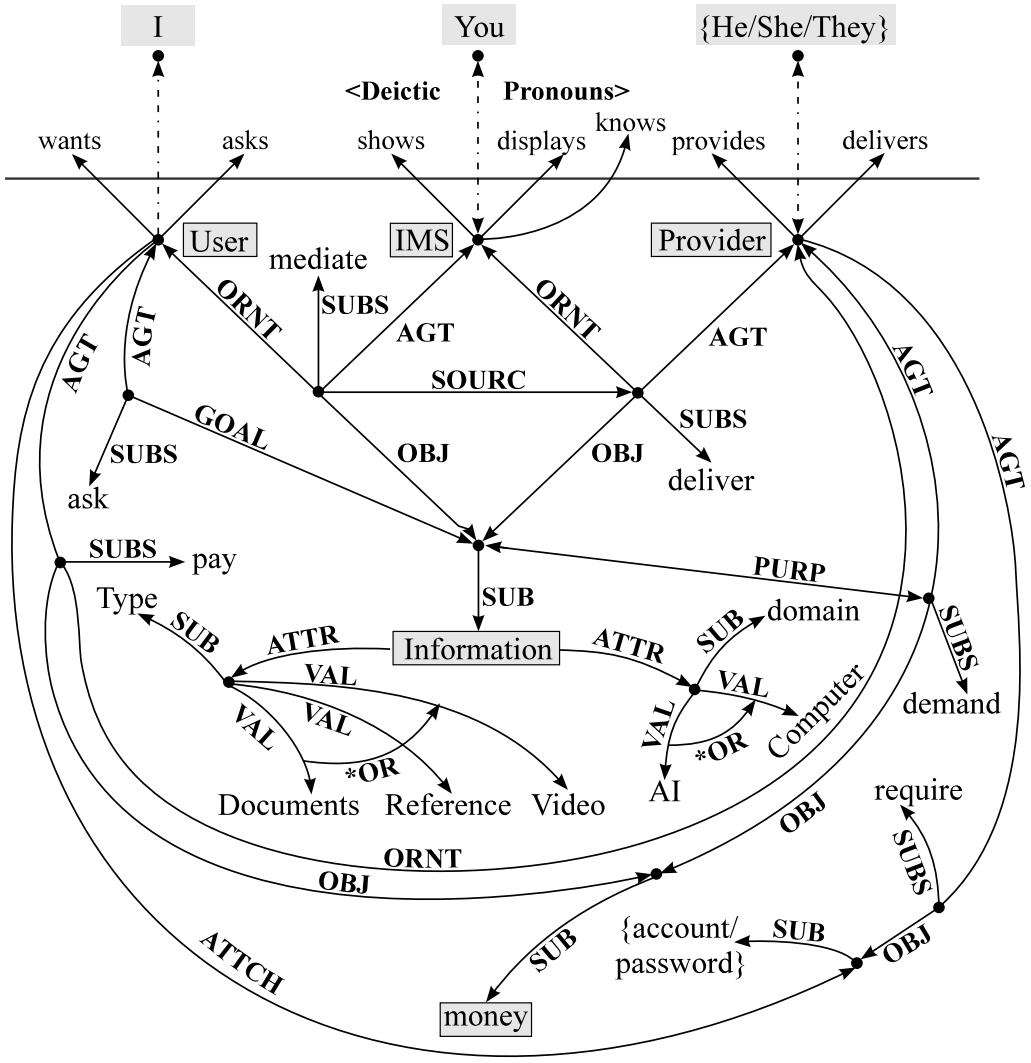


Figure 2: Typical information stored in the dialogue model

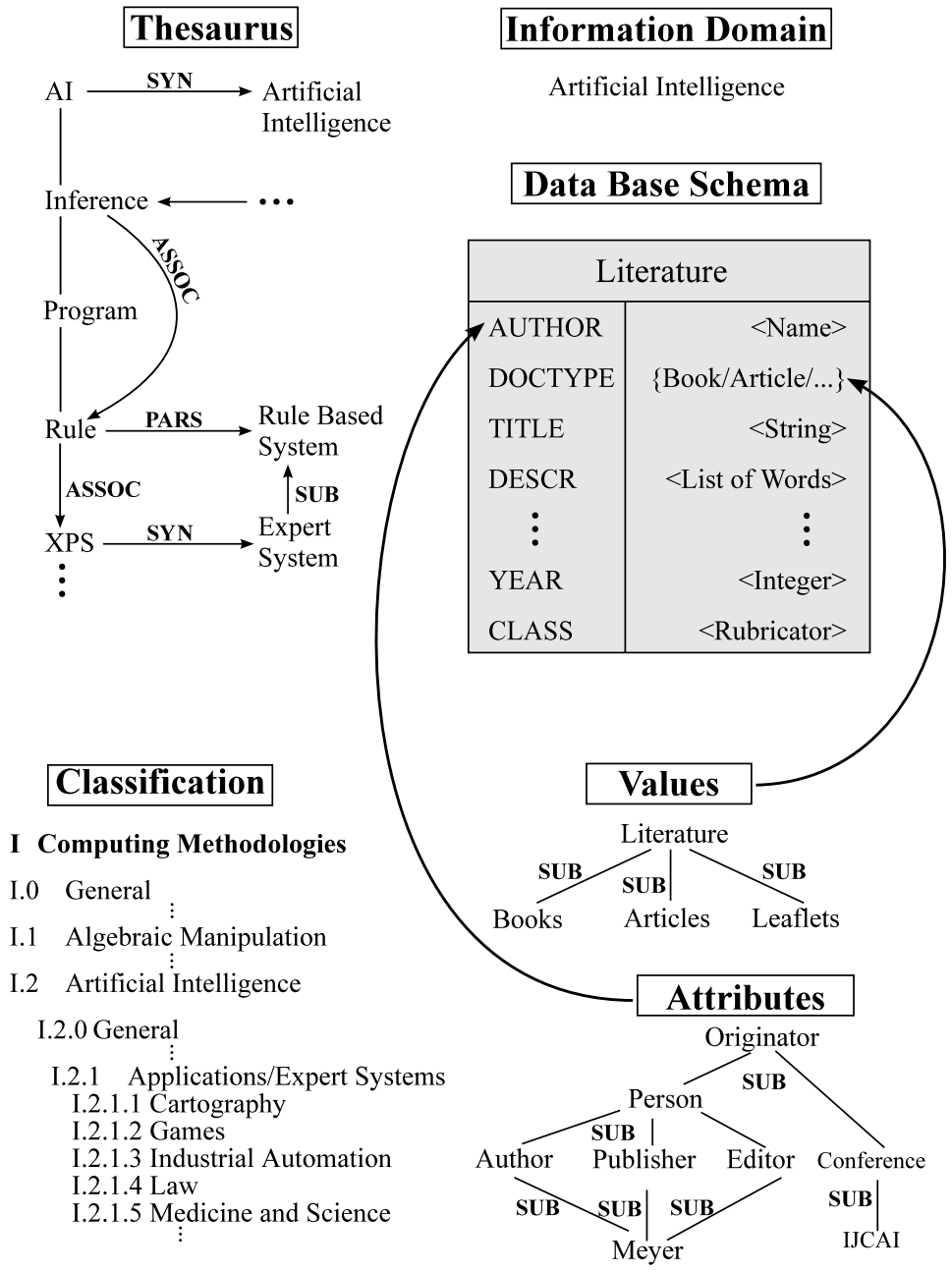


Figure 3: Background knowledge characterizing the information content

SYN - synonymy, ASSOC - association, PARS - part/whole relation and others). The entries of the thesaurus are the potential keywords usable in the description of documents in the data base (cf. attribute DESCR in the data base schema of fig. 3). The second component often met in bibliographic information systems is the classification system. It comprises a hierarchically organized system of non-lexicalized concepts associated with conveniently chosen rubricators defining a so-called decimal classification of informational subfields. One typical example — a part of the so–called CR–Classification [6] — is shown in the lower left side in figure 3. The entries of the classification system can be used additionally or alternatively to the descriptors (attribute DESCR) for the characterization of the documents in the data base (cf. attribute CLASS in the data base schema of fig. 3).

4. Organizational and commercial information. For the sake of completeness we have to mention a fourth part of background knowledge which concerns the technical and economical aspect of the communication with the provider. This part comprizes information about access rights, accounting, passwords and so on which rule the physical access to a special data base as well as information about costs and modalities of payment in the case of communication with commercial data bases. This aspect will not be deepened in this paper.

Summarizing, we can state that the dialogue model (1) is needed to find the appropriate information provider for a given user demand and to discern between dialogical or rhetoric phrases within the user queries and their informational kernel proper. The meta-knowledge about the content of the target data bases (2) and the classification systems (3) are important for the transformation of the user query into expressions of the retrieval language of the provider DBMS. From this point of view, the transformations Tr described in chapt. 5 realize a binary function $Tr : Q \times M \rightarrow R$ mapping queries Q of the user by means of the meta-knowledge M into the retrieval language R of the target system. Finally, the organizational and commercial data (4) are necessary to manage the transactions between end-user and information provider and thereby to survey the observation of financial, juridical and technical agreements.

4 The internal representation of the user query

This chapter deals with the problems connected with the "inner representation" of user demands and their translation into the retrieval language of typical target systems. According to the different ways of formulating a user query we shall divide our considerations into two parts. In the case of natural language access, semantic network concepts are applied to represent formally the content of the question. In the other case an internal retrieval language (IRL) is used for this purpose.
Nevertheless, there are some common features concerning the general structure of the transformation procedures which are similar in both cases. The main steps in the transformation process are illustrated in figure 4.

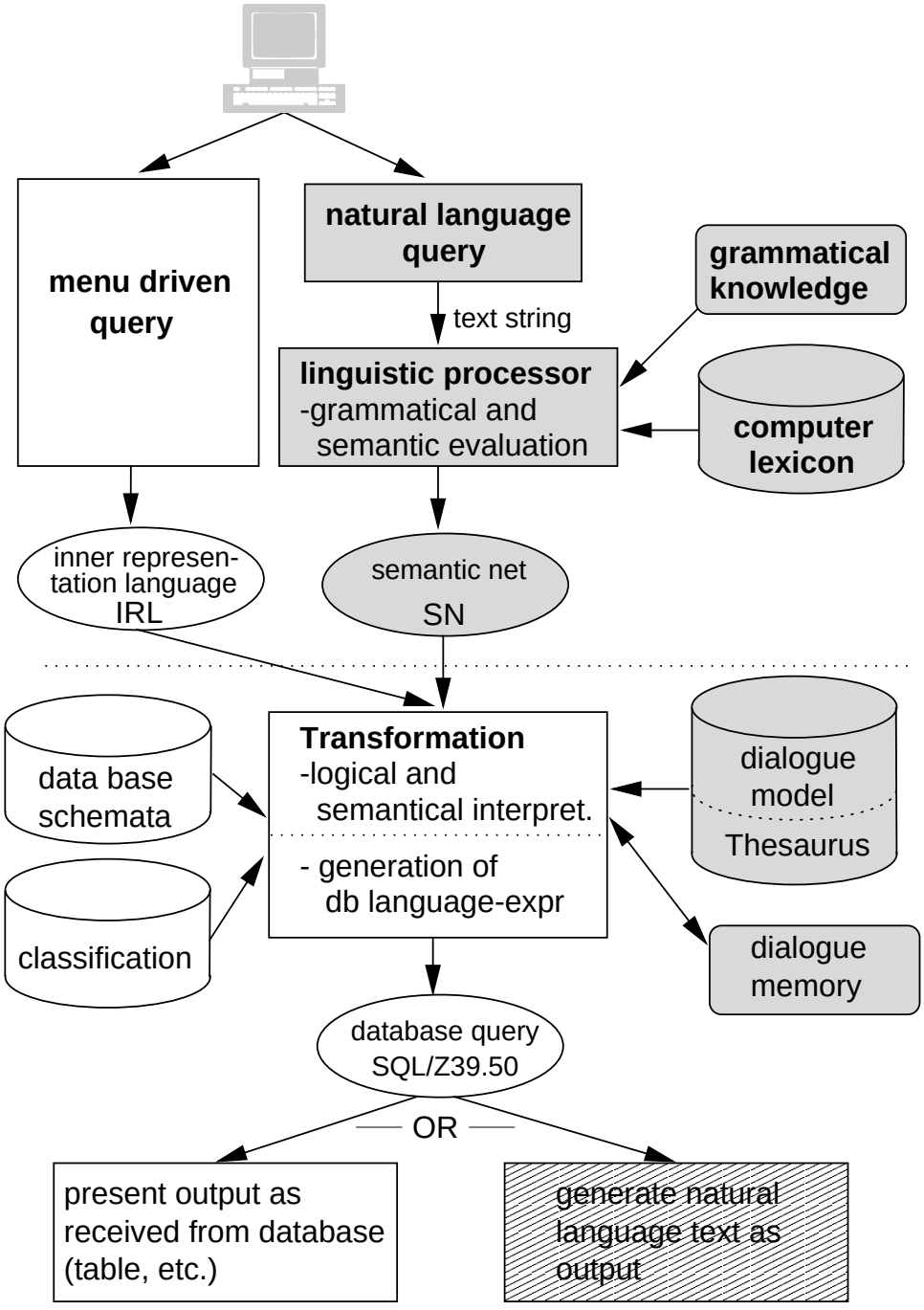


Figure 4: The general process of transforming a query

4.1 The representation of form oriented queries

A query specified by different terms in a form like manner (shown in fig. 5) is translated into the internal retrieval language IRL. This inner representation follows the pattern of a possibly large class of target languages. Because of that, the transformation steps needed for Method I are not so complicated compared with natural language input.
The IRL expressions are essentially constructed in terms of attribute–value–pairs defined by means of regular expressions:

```
<query>              ::=  (<expression>)
                          | (<expression> [<and> <expression>]*)
                          | (<expression> [<or> <expression>]*)
  <expression>       ::=  [not](<attribute> <rel> <value> [<vagueness>])
    <rel>            ::=  = | < | > | <= | >=
    <attribute>      ::=  <s_attribute> | <n_attribute>
      <s_attribute>  ::=  AUTHOR | TYPE | SUBJECT | DATE | PUBL | ...
      <n_attribute>  ::=  YEAR | MONTH | ...
    <value>          ::=  "<string>" | <number>
      <string>       ::=  [0-9a-zA-Z-_.!?&%]*
      <number>       ::=  [-]?[0-9]+[.]?[0--9]*
  <vagueness>        ::=  [<vag_value>]?
    <vag_value>      ::=  similar | optional | uncertain
```

In this definition an infix notation is used because several target languages, as for instance SQL, apply it too. The language defined above allows to express the vagueness of special terms. If no vague terms are contained in a query, all terms have to be dealt with as reliable information.

4.2 The semantic structure of the natural language query

When using the natural interface method, the transformation of the user query is carried out in two distinct steps. First, a linguistic processor analyzes the natural language expression from a grammatical and semantic point of view, thereby producing an intermediate semantic representation. This representation in its turn is processed by a transformation module which generates a query in the language of the underlying database system (which will be SQL in our example).
The semantic representation language describes the relations between the concepts contained in the natural language query. Each concept is represented as a node in a semantic net (SN) whose expressional means are taken from the MESNET paradigm described in [7]. A fixed set of predefined relations and functions (represented as edges in the SN) is used to describe the interrelationships between two (or more) nodes.
To give some idea of the semantic representation principles, we will discuss the meaning of the semantic representation of the user query shown in figure 1.
The overall structure of the semantic representation can be seen as a bracketed structure containing relational and functional terms as subexpressions. Nodes are either labelled by artificially created symbols like *G01, G02* or named after the corresponding concept, if such a concept/name mapping is unique.

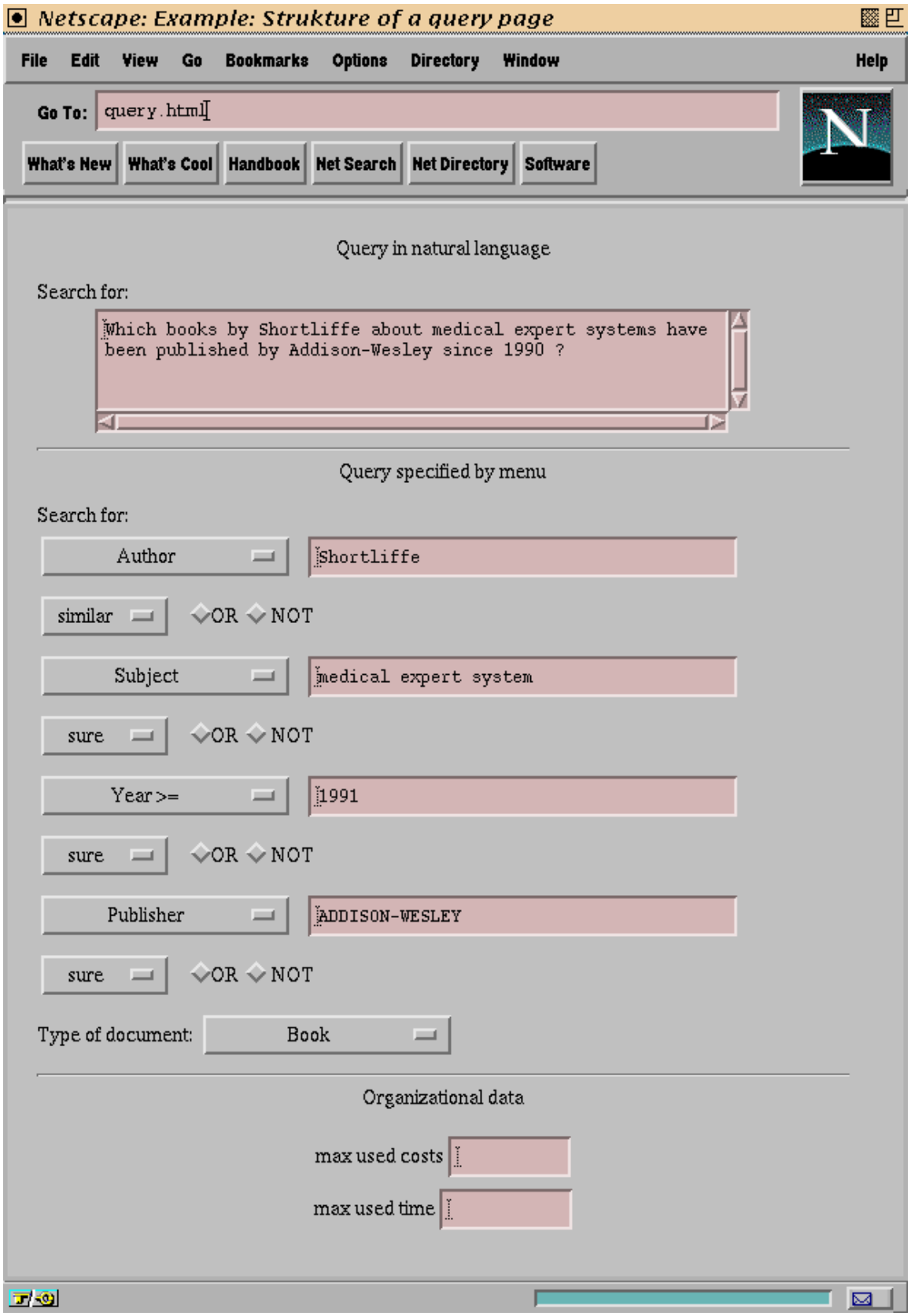


Figure 5: The html query page

In the following, a summary of the relations and functions used throughout the example is given. Functions are discerned from relations by putting an asterisk ("*") in front of their name. In the semantic net, a relation (REL a b) will be represented as an edge labelled "REL" between two nodes "a" and "b".

(TYPE t)	specification of the question type
(REF *val*)	reference to an antecedent node
(FOCUS n)	node n is the primary target of the query

(AGT h a)	node a refers to an agent who is performing some action h.
(MCONT s i)	node i refers to the mental content of a situation s.
(OBJ s b)	b refers to an object which is involved in a situation s, but is not physically affected by s.
(ORIG o_1 o_2)	o_2 describes the origin of o_1.
(STRT s t)	the situation or action s starts at the time given by node t.
(SUB o_1 o_2)	object o_1 is subordinated to object o_2.
(SUBS s_1 s_2)	s_1 is subordinated to s_2, with s_1, s_2 describing situations or actions.

(*PROPA p o) = c	construction of a new node c by modification of an object o with an associative property p

5 Transformation

The transformations of the user queries have to be differentiated according to the two principal methods:

Method I: using a form with different fields as question pattern;

Method II: allowing for natural language input for the user query.

To illustrate the process of transforming queries, the following example sentence will be used:

> Which books by Shortliffe about medical expert systems have been published by Addison-Wesley since 1990?

Having started his/her WWW client, the user essentially receives the page shown in fig. 5, but with empty input fields.
Employing the Method I, first the attributes have to be chosen from a pull down menu showing up on the screen after pushing the 'attribute' button. Then the user has to enter the attribute values into the corresponding value fields. Figure 5 shows a form which is already filled in (the entries in the fields above and below the horizontal line have to be understood as alternatives).
In the second Method the query is accepted in natural language which can be typed into the corresponding input field above the horizontal line.

5.1 Form oriented query

In the case of a form oriented input, it is necessary for the user to structure his query intellectually and to enter the appropriate terms into the masks defined by the menu. Thus, a part of the transformation process will be passed to the user.
The prescription of the masks can be a drawback, because there are possibly some demands the user would like to specify, which he isn't able to represent within the given form.
In our example we especially asked for books, a value which the user can specify but which perhaps will not occur in the data base. In the worst case, if there is no additional background information, the user gets no information at all. In the other case, if the transformation procedure has access to a predefined concept hierarchy where the concept "book" is subordinated to the concept "literature", and the superordinated value "literature" is integrated in the query, the user will get too much information.
Another problem arises from the specification of proper names or more general from uncertain information. If the name of the author isn't known exactly, the user must be able to express this deficiency. In Method I, this can be implemented by offering a pull down menu, where the user has to select the proper characterization of the vagueness of a certain term. At least the following characterizations of vagueness should be accepted by the interface:

- **sure** – there is no doubt about the specified value (especially not about its orthography); this is the default value.
- **similar** – it is possible that the specified value differs from the exact notation because of orthographical uncertainty, e.g. "Shortliffe" vs. "Shortlife").
- **uncertain** – it can be a completely different term which describes the informational aspect the user has in mind.
- **optional** – the user isn't sure if the corresponding attribute occurs in the database at all.

After completing the specification of the query by means of a form, the query has to be translated into the unified internal language IRL. The use of this intermediate language is particularly advantageous when the IMS is supported by an assembly of brokers mediating the communication between user and information provider (as it is the case with the MEDOC system [4]). In this case, the IRL not only defines the interface between user and provider but it is also used as means of communication between different brokers.
The translation of the query can be supported by means of CGI–scripts, where every input field is evaluated immediately after entering a new item into that field. The information specified will be integrated successively into the inner representation. During this procedure the logical operators have to be taken into account. They are inserted into the formal IRL expression according to the syntax given in chapter 4.1. After evaluating the input form, the query will have the following structure[1]:

[1]The specification of the classification attribute CLASS will be omitted here for the sake of brevity

```
( (TYPE = "book") AND
  (AUTHOR = "Shortliffe") AND
  (PUBL = "Addison-Wesley") AND
  (SUBJECT = "medical expert system") AND
  (DATE >= 1990) )
```

To bridge the gap between IRL and the retrieval language of a target system, the IMS in general has to take into consideration the synonymy between the attribute names of both languages. Let's assume the user wants to express that the man who wrote a book is named "Shortliffe". If the form has an input field named "author" and the target database contains only attributes like "writer" or just an abbreviation like "auth", there has to be a mapping between the form attribute (e.g. "author") and the target attribute (e.g. "auth") which can be used by the transformation process.
For that purpose, we can use the information contained in a thesaurus. In the case of abbreviations, the transformation process has to be supported by translation rules.
There are also difficulties connected with conceptual hierarchies of attributes (cf. fig. 3).

- If somebody tries to find a special author, he can ask the system the example query with *Author = "Shortliffe"*. If the data base contains no attribute "Author", two possible cases have to be dealt with.

  1. A parent node in a given attribute hierarchy can be taken as an appropriate attribute name; according to figure 3 this is "Person". In this case, the user gets more information than he really wants.
  2. If there is no superordinated attribute, the corresponding term in the query has to be dropped and the user gets all documents without consideration of the attribute "Author".

- The specification of "Person" shows the inverse problem. If the query asks for *Person = "Shortliffe"* and the data base contains the attributes "Author", "Editor" and "Publisher", subsumed by "Person" (cf. fig.3), the transformation has to build a disjunction of three expressions: `(Author = "Shortliffe")` $\vee$ `(Editor = "Shortliffe")` $\vee$ `(Publisher = "Shortliffe")`.

  The general rule to be used with attribute hierarchies (cf. lower right side of fig. 3) has the form:

```
(ATTR_N = V) =>

(ATTR_N+1(1) = V  OR
 ATTR_N+1(2) = V  OR
 ...
 ATTR_N+1(n) = V)
```

The analogue situation can be met with values. For example, if somebody searches for document type "literature" having in the data base scheme only the three fixed values "Books", "Articles", "Leaflets" which are subordinated to "Literature" (cf. fig. 3), the query has to be transformed in the following way:
The original query in IRL:

```
( (TYPE = "literature") AND
  (AUTHOR = "Shortliffe") AND
  (PUBL = "Addison-Wesley") AND
  (SUBJECT = "medical expert system") AND
  (DATE >= 1990) )
```

The transformed query:

```
( ( (TYPE = "book"      ) OR
    (TYPE = "article"   ) OR
    (TYPE = "proceedings")
  ) AND
  (AUTHOR = "Shortliffe") AND
  (PUBL = "Addison-Wesley") AND
  (SUBJECT = "medical expert system") AND
  (DATE >= 1990) )
```

After solving the attribute transformation problem, the query has to be translated into the target language.
In the last chapter, we have concentrated on SQL as a target language, in the following we want to discuss the translation of the user query into the Z39.50 protocol, which is a standard client/server protocol for information retrieval systems [8]. In contrast to SQL, there is no need for an extra hierarchy of attributes if one uses the Z39.50 protocol. In the often applied bib-1 attribute set associated with Z39.50, the attribute "Author-name" can have the following meanings: "person", "corporate author" or even "conference name". For more details see [8].
For transforming the query with the help of the Z39.50, a static transformation table will be sufficent. A table for the bib-1 attributes has the form:

```
AUTHOR          : Author-name
PUBL            : Name-publisher
SUBJECT         : Subject
DATE            : Date-publication
```

The Z39.50 is a session oriented network protocol, where the query has to be sent to the data base during an open session. Every message transferred within the session is called a PDU (Protocol Data Unit).
A search query within the Z39.50 consists of several so-called *rpn* (reversed polish notation expressions) connected by logical operations. The *rpn*-expressions are defined by the syntax shown in figure 5.1.
Here, the meta-symbol <RELATION> indicates one of the comparison operators admissible in Z39.50 which can be derived from the inner representation language IRL by means of special translation rules:

```
searchRequest =
{  smallSetUpperBound = 1
    ...                              ;;some more SR/Z39.50 specific terms
   query =
   {  type-1 = {
         attributeSet = OID 1.2. ....
         RPN = { rpnop =
                 {  RPN1 =
                    {  op =
                       {  attrTerm =
                          {  attributes =
                             {  relation = Equal
                                    use = Name-publisher
                             }
                             term =
                             {  general = "Addison Wesley"
                    }  }  }  }

                    RPN2 =
                    {  op =
                       {  attrTerm =
                          {  attributes =
                             {  relation = greaterThanOrEqual
                                    use = Date-publication
                             }
                             term =
                             {  numeric = 1990
                    }  }  }  }

                    op = { and }
               } }
           ...
}  }
```

Figure 6: An excerpt from the Z39.50 query

```
rpn =
{  op =
   {  attrTerm =
      {  attributes =
         {  relation = <RELATION>
                 use = <ATTRIBUTE_N>
         }
         term =
         {  <VALUE_TYPE> = <VALUE>
}  }  }  }
```

Figure 7: A general *rpn*-expressions

```
RELATION("=")  := "equal"
RELATION("<")  := "lessThan"
RELATION(">=") := "greaterThanOrEqual"
```

The "term" <VALUE_TYPE> = <VALUE> specifies the value of <ATTRIBUTE_N> together with the value domain belonging to this attribute. <ATTRIBUTE_N> characterizes a special attribute from depth N within a given attribute hierarchy supported by the Z39.50 protocol.
After the transformation, the query has the structure shown in fig. 5.1[2]
This query is written in the ASN1 (ISO 8824), the Abstract Syntax Notation, a high–level data structuring language belonging to the Z39.50 protocol.
This notation has to be translated again, this time using the BER (Basic Encoding Rules) which is defined by another standard, the ISO 8825.
After completing the corresponding transformations, the query has a numerically encoded form:

```
\270\020\237\037\007 ... ,
```

which can be sent to the Z39.50 server.

5.2 The natural language query

After entering the natural language query, the linguistic processor translates it into the intermediate semantic representation; see figure 8 for the semantic net corresponding to the example query used throughout this paper. The transformation takes place after the linguistic processor has done its work. At this stage, the basic information about the query is already made explicit in the semantic net.
For example, the linguistic preprocessor has already determined on which object the query is focussing. Also the interrelationships between the components of the query

[2]Only a few typical parts of the query structure are described in the example because the whole expression is too long to be printed here.

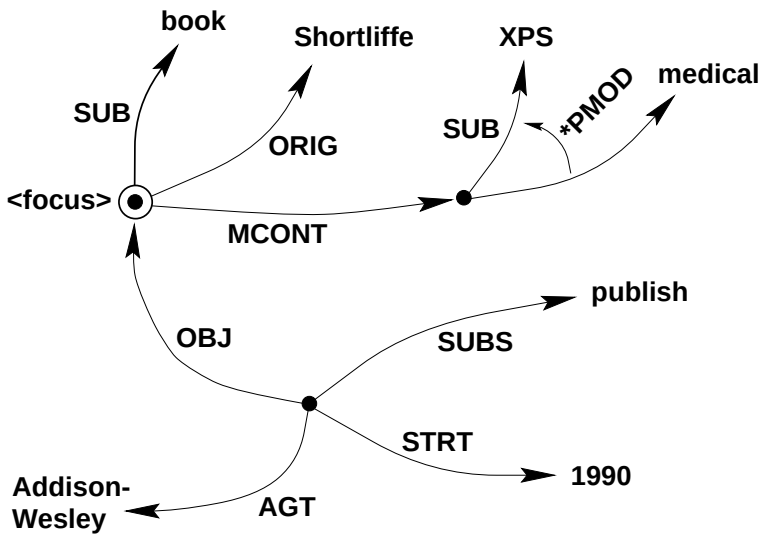


Figure 8: The semantic net representation of the example query

have been translated into the relational structure of the semantic net. The node labelled <*focus*> represents the focus of the query.
The transformation module's task is to determine the exact pragmatic interpretation of the SN using the background knowledge of fig. 9. At the beginning of this interpretation process, the edges directly connected to the focus node of the query are examined. Following the 'SUB' edge in the example from fig. 8, the transformation process gets more detailed information about the subject of the query, which is subordinated to the concept "book" in this case, and which results in (Doctype = Book) because of rule *r9*.
Assuming that the background knowledge shown in fig. 9 is available, it can be deduced that the 'ORIG' (origin) edge must either point to an "author" or an "editor" (according to rule *r1*). If it is known that "Shortliffe" qualifies as an author, an `(Author = "Shortliffe")` part of the SQL query can be derived using rule *r4*; otherwise a disjoint term has to be generated according to rules *r1, r4,* and *r5*.
Similar conclusions can be drawn for the 'MCONT' edge. Here the system finds out that the mental content of a book is related to some subject information (rule *r2*). As a consequence, applying rule *r6* yields "Descr" or "Class" as a matching attribute for the subject of the book. This leads to the expression `(Descr = "medical XPS")` $\vee$ `(Class = "I.1.2.1.5")` in the resulting query. The built-in function f realizes a mapping from the verbal description "medical XPS" into the rubricators of the CR-classification.

After having outlined the general idea of the transformation process, we would like to point out some additional difficulties which have to be taken into account when performing the transformation.
For example, a problem likely to arise is that the user is paraphrasing some elements of his query, e.g. he might say "medical expert system" as well as "expert system for medicine". The semantic equivalence of such paraphrases is automatically discovered during the semantic interpretation by using so-called meaning postulates. In some

Rules:

(Convention: Lower case symbols denote variables. All variables with the same symbol are instantiated with the same value throughout the rule set.)

r1: (q SUB BOOK) ∧ (q ORIG a) → (a SUB AUTHOR) ∨ (a SUB EDITOR)

r2: (q SUB BOOK) ∧ (q MCONT s) → (s SUB SUBJECT)

r3: (q SUB BOOK) ∧ (o OBJ q) ∧ (o SUBS "publish") ∧ (o AGT p) → (p SUB PUBLISHER)

r4: (a SUB AUTHOR) → {SQL-Expression: *"Author = a"*}

r5: (a SUB EDITOR) → {SQL-Expression: *"Editor = a"*}

r6: (s SUB SUBJECT) → {SQL-Expression: *"Descr = s"* ∨ *"Class = f(s)"*}

r7: (p SUB PUBLISHER) → {SQL-Expression: *"Publisher = p"*}

r8: (o SUBS "publish") ∧ (o STRT t) → {SQL-Expression: *"Year >= t"*}

r9: (q SUB BOOK) → {SQL-Expression: *"Doctype = 'BOOK'"*}

Facts:

("Shortliffe" SUB AUTHOR), ("Myers" SUB AUTHOR), ...

Figure 9: An excerpt from the transformation knowledge base

cases, also different relations may be selected from the linguistic preprocessor for a certain component, e.g. choosing "ORIGL London" instead of "ORIG London", because there is possibly no background knowledge to decide if London is a location (ORIGL – local source) or a human being (ORIG – mental source). Such ambiguities are unavoidable since some relations are difficult to distinguish automatically. This effect can be covered by adding disjoint expressions `(ORIGL q a)` ∨ `(ORIG q a)` to the query.
Another problem which has to be taken into account arises from synonymous words like "author" and "writer". Since the data base scheme normally contains only one word for a special attribute in the actual database, the transformation module needs to consult a thesaurus if it cannot match a given query element with a database attribute. Using a thesaurus is also beneficial when there is some preciseness lacking in the query. If the user asks about a medical XPS, but there is only data available for rule-based medical systems, then the transformation system could infer from the thesaurus (see fig. 3) that XPS are actually subordinated to rule based systems and deliver a valid answer nevertheless.

Altogether, the rule applications outlined above give the SQL expression for the sample query is shown in figure 5.2.
Finally, it should be mentioned that the transformation is also assisted by a dialogue memory, which remembers the semantic structure of the most recent queries from the user. Information from the dialogue memory is especially useful when the user is entering incomplete or elliptical sentences like *"and which before 1990 ?"*. If this sentence is following our initial example (" *Which book by Shortliffe...*"), then the system

```
SELECT * FROM Book
WHERE Doctype = "Book"
  AND Author = "Shortliffe"
  AND ((Descr = "medical XPS") OR (Class = "I.1.2.1.5"))
  AND Publisher = "Addison-Wesley"
  AND Year >= 1990
```

Figure 10: SQL expression for the sample query.

can fill in the missing attributes like "author = Shortliffe" etc. from the foregoing question in the dialogue memory.

6 Present state and future development

From the two methods described in the paper, natural language access to IRS and form-driven dialogue, the first (Method I) is followed in the project MEDOC sponsored by the Federal Ministry of Education and Research (BMBF). One important aim of the MEDOC project is to provide an intelligent brokering system to support the user in finding the needed information in a world-wide distributed and networked system of information sources.
The second way (Method II) is adhered to the system LINAS (**Li**teraturrecherche in **na**türlicher **S**prache) of the FernUniversität Hagen. A first prototype of the natural language interface of the system is already used in connection with the target IRS AIDOS for managing and accessing the AI literature of the local library held in the AI lab.
It is planned to merge these different approaches into one system (i.e. into the above mentioned system LINAS) therewith combining the advantages of natural language communication with IRS and the strategically important access support to information systems distributed in international computer networks.

References

[1] Helbig, H. et al.: "The natural language interface NLI-AIDOS"; in J.Newer Gener. Computer Syst. 3 (1990) 3; Akademie-Verlag, Berlin; p. 221-246.

[2] Fuhr, N.; Großjohann, K.: "Broker Architecture"; MeDOC internal report; `http://ls6-www.informatik.uni-dortmund.de/projects/MEDOC/broker/intern/architecture.ps`.

[3] Helbig, H.; Mertens, A.: "Der Einsatz von Wortklassen für die automatische Sprachverarbeitung"; Informatik Berichte 158; FernUniversität Hagen; 1994, Teil 1; Überblick über das Gesamtsystem; chapter 2.1, p 4 ff.

[4] Brüggemann–Klein, A. et al.: "Entwicklung und Erprobung offener volltext–basierter Informtionssysteme für die Informatik"; 1995;
http://medoc.informatik.tu-muenchen.de/ovid_1.ps.

[5] CNIDR: "The scientific and technical attribute and element set";
http://stas.cnidr.org/STAS/.

[6] ACM: "Computing Classification System"; (formerly Computing Reviews)
http://www.acm.org/class/.

[7] Helbig, H.; Herold, C.: "MESNET A Multilayered Extended Semantic Network"; Informatik Berichte 185; FernUniversität Hagen; 1995
http://voss.fernuni-hagen.de/gebiete/pi7/papers/ibmesnet.ps

[8] ANSI–NISO: "Information Retrieval: Application Service Definition and Protocol Specification"; ftp://ftp.loc.gov/pub/z3950/offical/part1.ps - part4.ps