EXPRESS: An Experimental Interface for Factual Information Retrieval

Heinz Ulrich Hoppe, Karin Ammersbach, Barbara Lutes-Schaab, Gaby Zinßmeister
GMD-IPSI, Dolivostr. 15, D-6100 Darmstadt (FRG)
e-mail: hoppe@darmstadt.gmd.dbp.de

Paper submitted to SIGIR 90

Abstract

The EXPRESS system has been designed and implemented in order to explore methods for user assistance in accessing complexly structured factual databases, e.g. relational product databases. Terminological support in this area has to take into account that different controlled vocabularies may be used in a variety of attributes spread over several relations. In our approach, traditional thesaurus structures are extended in order to cope with these problems and to encode further domain-specific knowledge. User support in query reformulation is based on this enriched thesaurus as well as on the local evaluation of the retrieved data sets. Concepts for the representation of retrieval strategies in the form of plans and their potential use in future systems are discussed.

1 Introduction

Intelligent interfaces for Information Retrieval (IR) can be based on different sorts of knowledge, such as knowledge about the user and his or her information need, expert strategies and tactics for query planning and reformulation, terminological knowledge, or even content-oriented, semantic descriptions of the objects gathered in the database. Significant progress in supporting direct end-user access to public databases may be expected from any of these knowledge bases and it is therefore desirable that an intelligent retrieval system should use all these in an integrated form. On the other hand, none of the different approaches mentioned is already able to offer well-established and easily applicable engineering methods in order to incorporate the respective features into a retrieval system. There are still open research issues in the different relevant fields. In spite of integration being desirable, significant advances of the basic mechanisms towards the development of a practical methodology may also be achieved through a "divide and conquer" strategy, i.e. by means of elaborating on a subset of the approaches separately.

A considerable amount of work in the area of intelligent retrieval interfaces focuses on user modeling (e.g. Brajnik et al., 1987; Brooks et al., 1985). There are different notions of user models (cf. Kobsa, 1989), which are all relevant to IR. Rich's GRUNDY system (Rich , 1979) is an early example for user modeling based on stereotypes. The stereotype approach views a specific user as a representative of a category or class which is predefined in terms of several long-term characteristics and typical preferences. In its simplest form, such an approach can

hardly be flexible enough to capture individual differences. More flexibility is achieved by allowing individual refinements of predefined prototypes and multiple inheritance from different classes.

A more fundamental problem with the use of stereotyped user models in IR systems originates from the possibility that one and the same individual may show qualitatively very different information needs over time, even within one session. There is a dynamic interaction between the problem context which gives rise to the information need, the user's role in this problem context, and the user's articulated problem description. In many cases, the information need is much less determined by the relatively permanent individual characteristics of a user than by the problem context or task. Task modeling in its different forms (e.g. Card et al., 1983; Payne & Green, 1986; Hoppe, 1988) can be used to represent fixed operational schemata which apply to certain problem classes. Request of some specific information may be one step in such an operational schema for attaining a task such as e.g. travel planning. Inside the information retrieval task, operational schemata have also been identified in terms of search tactics and strategies (e.g. Bates, 1987; Fidel, 1985). Existing task modeling approaches are not directly applicable in order to represent tactics and strategies in IR, because here the final actions are not yet determined when "the procedure is entered". This is due to the fact that "goals" in IR cannot be simply defined in terms of state changes in the underlying system, but have to be regarded as changes in the user's knowledge state. Therefore, a continuous evaluation of system feedback is necessary in order to pursue a certain strategy and determine the next action. One of our current research goals is to combine user support mechanisms based on task models with more flexible planning methods (e.g. Hayes-Roth & Hayes-Roth, 1979; Mannes & Kintsch, 1989).

The notion of retrieval tactics and strategies constitutes a specific aspect of the expert system approach to intelligent IR (cf. Brooks, 1987), since these are typical components of the expertise provided by professional search intermediaries. The expert knowledge encoded in intelligent retrieval interfaces also includes simple procedures (e.g. connection to the host) and conversion from a standardized representation of Boolean expressions to the specific query languages. For this part of the job, there are already acceptable engineering solutions. More "intelligent" features comprise the elaboration of a Boolean query from an unstructured list of natural language terms (as e.g. in EP-X, cf. Smith et al., 1989) or from a partial analysis of free natural language input (as e.g. in PLEXUS, Vickery & Brooks, 1987) as well as the construction of a plan for the incremental evolution of queries, as done in EURISKO (cf. Barthès & Glize, 1988). A common shortcoming of this category of intelligent retrieval interfaces may be attributed to the paradigm of simulating the reasoning of a human expert. Such systems (like human intermediaries) are usually not equipped with mechanisms for performing exhaustive analyses of the retrieved data sets. Although the analysis of term frequencies in given response sets is already supported by existing technology (for example in the retrieval languages MESSENGER and QUEST), the logical next step of using regularities in a given data set as clues for system-supported query reformulation is generally not taken, e.g. to assist the user in broadening as well as narrowing or a change of focus. One exception is the EUROMATH interface for bibliographic retrieval in the domain of mathematics (McAlpine & Ingwersen, 1989). This interface provides access to the results of host frequency analyses as additional information for the user; however, it does not utilize them in its internal search strategy.

Most of the existing intelligent retrieval interfaces support only the retrieval of bibliographic references or full text documents. In these areas, terminological knowledge is available in the form of thesauri, which can be used as additional knowledge sources. Traditional thesauri have been replaced by richer knowledge structures, such as semantic networks or frames (Monarch & Carbonell, 1986; Shoval, 1983; Smith et al., 1989). Accordingly, indexing is seen as a semantic representation of the document content. The issue of "natural" or commonsense semantics in user utterances or texts leads to open problems in AI and computational linguistics. In order not to overload the strive for intelligent IR with these "heavy" problems, we consider it appropriate to further exploit the notion (and use) of thesauri as terminological knowledge bases. This seems to be particularly valid in the somewhat neglected, but practically very relevant domain of factual databases, such as chemical or materials databases. In these areas there is a clearly defined technical terminology. A deeper understanding of this terminology usually requires a thorough scientific background. It is questionable if an attempt should be made to provide the retrieval interface with this kind of deep knowledge, as long as there are open problems which can be solved on the terminological level. Using current technology, information about materials or chemical substances can be adequately stored in relational format, each attribute representing a particular feature expressed in terms of a numerical value or range, a formula, or a textual description. We will show that access to this kind of information system can be supported by an enriched thesaurus which contains not only taxonomic but also domain-specific relations and reflects the attribute structure of the underlying relations by means of different facets.

Based on this critical view of existing approaches to intelligent IR, we have focused our research on the following open problems:

- terminology support for information retrieval from complexly-structured factual databases,
- the implementation of query planning and reformulation mechanisms using this kind of terminological knowledge bases as well as mechanisms for an exhaustive analysis of the retrieved data sets,
- the relevance of task models and planning mechanisms for user guidance in IR.

In order to put our ideas into practice, we have implemented a prototype called EXPRESS (EXperimental PRototype for Exploring Support Strategies in factual IR). The following sections will be successively devoted to an overview of the functionality and architecture of EXPRESS at a global technical level, a brief description of the specific problems involved in fact retrieval, a structural description of the underlying terminological knowledge base, and the query evaluation and reformulation mechanism in its current form as well as envisaged extensions. Particularly in the latter aspect, we will assume a cognitive science point of view in that we regard information retrieval as a planning or problem solving activity.

2 The EXPRESS system

As a testbed for intelligent assistance in information retrieval, we have implemented a prototype system which supports users in accessing a factual database of products for wood protection. The EXPRESS (EXperimental PRototype for Exploring Support Strategies in factual IR) system provides terminological support during the process of (re-)formulating queries to satisfy users' information needs.

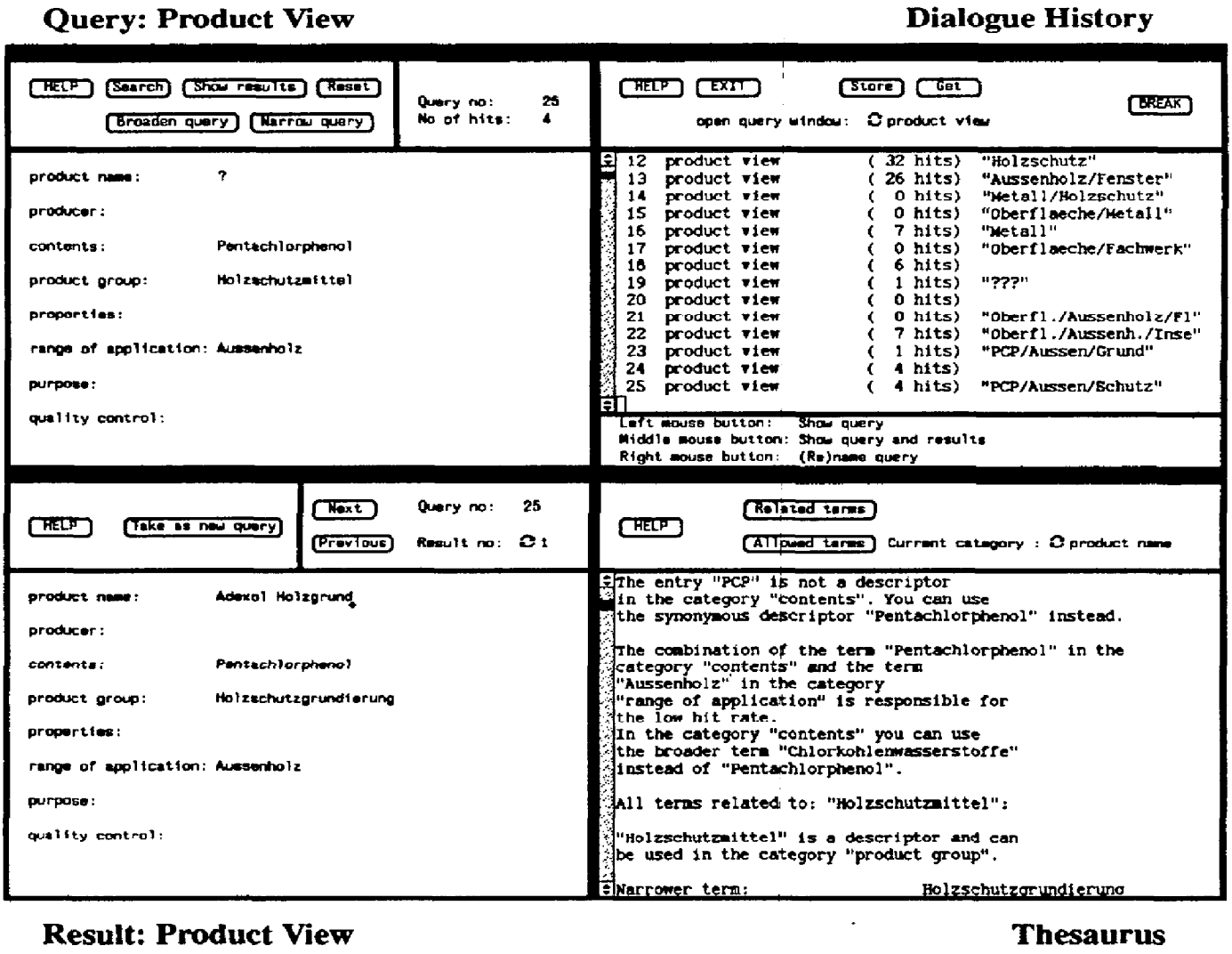

Figure 1 : The surface of the EXPRESS system

Figure 1 shows the surface of the system. In the upper left window, queries are constructed by filling out an onscreen form. Such a form represents a predefined view of the database. This can be seen as a simple **query-by-example** (QBE) interface as presented in (Zloof, 1983). At the moment, users may choose from two such views: the product view, which describes specific products; and the content view, which can be used to place queries concerning the ingredients of products and their potential effects. In implementing this type of access to the database, the intention was to avoid the typical problems end users encounter with any given command-oriented retrieval language.

Query results are then presented successively in the lower left window in a form similar to that of the query. Any given document in an answer set may be transferred to the query window and used as the basis for a new query (using the 'take as new query'-button).

In the window on the upper right the dialogue history is recorded. The user is able to assign meaningful catchwords to former queries. Following the paradigm of **'query-by-reformulation'**, former queries as well as answers can be used as a basis for developing new queries. This paradigm has been used in some other prototypes such as the HELGON system (Fischer & Nieper-Lemke, 1989).

One of the main problems of casual users of a retrieval system is to find the 'right' terms to describe their information need. The trouble lies in the discrepancy between the user's personal vocabulary usage and the terminology used to index the objects in the database. Therefore EXPRESS supports users in mapping user terms onto system terms. Once the (controlled) vocabulary has been found, the response to a query might nevertheless be unsatisfying in that the hit rate is too small or too large. To overcome these difficulties, the EXPRESS system offers help in the form of suggestions as to which terms in the query could be replaced during reformulation in order to arrive at a satisfactory answer set. In the current system version, the user judges whether or not a given answer set is satisfactory, and then explicitly invokes the broadening/narrowing algorithms to receive reformulation suggestions. The main knowledge base for (re-)formulation purposes is an **enriched thesaurus** implemented as a semantic network. The thesaurus is described in detail in Chapter 4, the **broadening** and **narrowing** functions are explained in Chapter 5.

A **'check-value' function** supports the mapping of terms on the level of morphological similarity. Starting from the user-given term T in a category C, the algorithm checks the following conditions in the given order and suggests the derived descriptor D and additional information depending on the valid case:

(1) Is T a descriptor D in the category C?

(2) Is T a synonym for a descriptor D in C?

(3) Is T morphologically similar to a descriptor D in C?

(4) Is T morphologically similar to a synonym S of descriptor D in C?

Steps (1) to (4) are then performed on all other categories with controlled vocabularies, which can, for example, result in the information that the requested term is a descriptor in a category other than the one in which it was requested. To check these conditions the algorithm uses the faceting in the thesaurus (see Chap. 4), the synonymy relationship, and a method for assessing morphological similarities between terms.

Apart from the help described above, where the system uses the thesaurus as a knowledge base to deduce the appropriate search terms, the user can **browse in the thesaurus** independently of the ongoing search. On the one hand, all the allowed terms for any category (attribute) can be looked at alphabetically. On the other hand, starting from a user term, EXPRESS will display all the information it can derive from this term using the thesaurus links and by means of the same

algorithms as mentioned above. I.e. synonyms, related descriptors in the same or other categories, and textual definitions of terms are presented in the thesaurus window.

The goal in designing the EXPRESS system was to explore ways to automate various user support functions. For purposes of experimentation and transparence, in the current version, all support mechanisms are semi-automatic, i.e. available upon user request, usually by means of a labeled button. This provides users with easy and direct access to all options and alternatives onscreen. To achieve this, we have used the top level control mechanism of event handling as offered by SunView. Thus, SunView events trigger the invocation of the functions performed by the underlying Prolog programs.

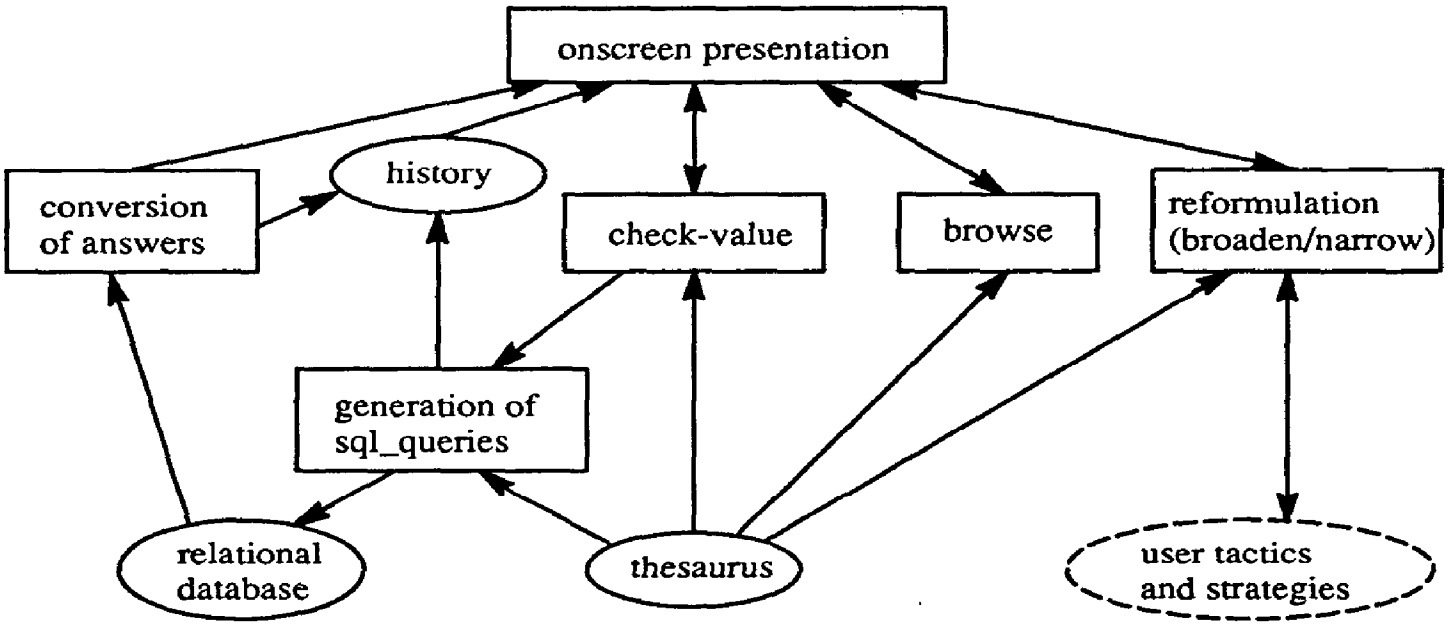

Figure 2 : The components of EXPRESS

In Figure 2 the system components are shown from the functional point of view. The ovals designate knowledge bases and the boxes the methods to be applied to them. In placing a query, the system checks the terms and generates an SQL-query which is sent to the relational database. The database response is converted into a representation which can easily be used in the onscreen, form-oriented presentation. At the moment, the reformulation functions are implemented implicitly in the program, each reformulation rule being a specific Prolog clause. In the future, they will be represented more explicitly to separate the inference engine from the knowledge base. The advantages are obvious: the knowledge base can easily be modified, extended and improved without changing the control mechanism. Beginning with a small knowledge base, one can experiment with the system and incrementally extend it. The other components of the figure should be clear from previous explanations.

The interface with its help facilities is independent of the specific data base. A second relational data base with SQL-access and an available thesaurus has been connected to EXPRESS as well. The database itself is implemented using Sybase; the thesaurus and the interface are implemented in Prolog; the system runs on Sun workstations.

3 Specific problems encountered in retrieval from factual databases

The choice of fact retrieval (e.g. in materials or product databases) as domain for the EXPRESS prototype poses a number of problems which differ from those encountered in bibliographic retrieval. Empirical evidence for some of these problems is provided by analyses of the behavior of professional intermediaries during their interaction with clients in online retrieval sessions using bibliographic and factual databases (Ammersbach, 1986; Ammersbach et al., 1988). The main problems observed and the way in which they are being taken into account in the EXPRESS interface are described in this section.

Attribute selection

In bibliographic databases, the structure of the records is straightforward and the categories used are clear, i.e. author, abstract, etc. The data type used to describe the categories is also generally intuitive; e.g. abstract and title are text strings, publication year is an integer, and so on. In product or materials databases, often a very large number of attributes (sometimes more than 100) and a variety of attribute types (e.g. intervals, integers, text) are used to describe each documented entity. This is a source of difficulty, since confusion may arise as to the meaning of the attribute labels (field names), and as to which attribute to search in for a given information need. The problems include nonintuitive field labeling and the fact that nuances of the same phenomenon may be described in different places in a given record. Both these characteristics lead to ambiguity and uncertainty as to where to search for what. In the EXPRESS system, we have partially countered this problem by using a QBE interface reflecting predefined views of a relational database, and by referral to an online thesaurus for an onscreen display of attribute values (see below).

Attribute-specific terminology support

Solving the problem of selecting the right attribute does not solve the terminological problems within the attribute. Searching in factual databases still shares many of the terminological pitfalls of bibliographic databases. Here, too, polysemy and synonymy abound, insofar as attributes are described using natural language terms. This is compounded by the fact that thesauri are often unavailable. In the case of attributes containing numerical values, users sometimes express their information need as a natural language circumscription of information which is numerically coded in the database. Another problem is that search queries may be posed in terms of different units of measure than those used in the database. This is especially true for attributes containing ranges of values (e.g. temperature range). The empirical data shows that especially in the latter cases, query reformulation often involves an iterative change of ranges or discrete values within the chosen attribute, and if the response set is still too small, a decision is made to switch to a different, related attribute, usually in hopes of broadening the search. The decision is based on the knowledge of the professional intermediary about cross-attribute relationships.

Thus, different attribute types require different kinds of terminological support. The provision of an online thesaurus which is referred to by the check-value algorithm already described helps

solve lexical problems and synonymy in natural language fields. In addition, EXPRESS offers pop-up menus for each attribute which display a partial alphabetical list of allowed terms, thus providing users with examples. A complete list of allowed terms for a given attribute is available in the thesaurus window. A planned rule-based expansion of the equivalency relationship to link natural language terms to numerical values and ranges of values and these to each other will alleviate the problems arising from different data types. The introduction of faceting and a cross-attribute relationship in the thesaurus, which is described in more detail in the next chapter, allows the simulation of expert knowledge in broadening and narrowing unsuccessful queries.

Precision-oriented search

Another empirical finding is that when describing their information need, clients of fact databases usually have a concrete application in mind which implies that the target of the search has to comply with more rigid constraints than is usually the case in literature search. In some cases this can even be a performance specification of a sought-for material, in contrast to the subject description typical of pre-search interviews during bibliographic retrieval. Still, due to the plethora of attributes and lack of controlled vocabularies described above, this does not mean that mapping the information need onto system terminology is easier. The formulation of a search query on the same level of specificity as the user's information need often leads to a very small or even null response set. Thus, the strategy of first employing the tactic of broadening is often used as an interim step to achieve a hit rate large enough to enable further narrowing in order to iteratively achieve the desired precision. EXPRESS supports both narrowing and broadening, described in Chapter 5.

4 The thesaurus knowledge base in EXPRESS

In the EXPRESS interface, the terminological knowledge base plays a central role in supporting the user during both initial query formulation and query reformulation. The thesaurus at the core of the terminology base provides a pool of networked terms whose various links are exploited by the check-value algorithm described in Chapter 2, and by the algorithms for broadening and narrowing the scope of a query as described in Chapter 5. It can also be easily referred to for browsing independently of a specific query. As a pragmatic departure point in designing the terminological knowledge component, we have chosen to enhance traditional thesaurus structure in compliance with the exigencies of the chosen domain of fact (here product) retrieval described in Chapter 3. In the following, the most important enhancements of conventional thesaurus structure and their relevance for the support algorithms are described.

Faceting for fact retrieval

The EXPRESS thesaurus device described in this section is a type of faceting intended to counter the problems caused by the large number of attributes used to describe any one documented entity in a factual database, as described in the previous chapter. The explicit assignment of each controlled term to a particular facet in the thesaurus is used by both the check-value and the broadening algorithms.

In bibliographic indexing and retrieval, the object to be described and retrieved is generally a document as a whole. Postable terms from a thesaurus, or descriptors, are often only located in a single category for thesaurus index terms. If a classification scheme or subject indexing scheme other than a thesaurus is used, these terms may also be located in their own fields. Still, the terms selected from each of the ordering systems refer to the entire document. To counter the fact that any object to be indexed can be described from various points of view, or aspects, the concept of a faceted classification was devised. A facet, in this sense, is a semantic cluster, i.e. a set of associated terms with a basic semantic affinity.

In a highly structured factual database, much of what is achieved in a thesaurus by clustering into facets has already been performed and is reflected in the fine structure of the records; i.e. various attributes (equivalent to facets) are used to describe the substance, material, object, etc. undergoing the documentation process. Each attribute is described using a set of terms or values which already share a semantic affinity, i.e. their membership in the semantic cluster represented by the attribute. In EXPRESS we have made this implicit semantic categorization explicit by partitioning the controlled vocabulary according to the attributes it is used for. Thus the facets in the thesaurus are derived from the underlying attribute structure of the database. We assume these classes to be mutually exclusive. Thus a separate thesaurus is implemented for each attribute, i.e. hierarchization takes place within the cluster allowed for each attribute, respectively. Any given postable term's membership in a particular partial thesaurus is indicated by a special thesaurus relationship named 'facet'. This means that when a term is looked up in the thesaurus, it becomes immediately apparent which attribute the term can be used to describe. This is a condition for the functioning of the check-value algorithm described in the previous chapter. The coding of each term with a tag indicating its facet (i.e. allowed attribute) is also a prerequisite for the cross-attribute relationship to be described in the next section.

In the product view of the current EXPRESS database, for example, eight attributes are used to describe each documented product: *product name, producer, contents, product group, properties, range of application, purpose,* and *quality control*. The choice of attributes as well as the terminology associated with each attribute were derived from standard technical specification sheets available from the manufacturers of the described products. With the exception of *product name* and *producer*, which are proper names, and of the freetext attribute *properties*, each of these attributes is terminologically controlled, and each allowed term is linked in the thesaurus to the attribute (facet) which it can be used to describe. Thus a thesaurus search for, for example, the term *insecticide* will reveal that this term can be used to instantiate the attribute *product group*; looking up *wood pests* will lead to the attribute *range of application*. The fact that these two terms, although in different facets, are obviously related, inspired the associative relationship described in the next section.

Cross-attribute relationship

One specific type of semantic knowledge possessed by experts in the domain of factual databases is that of likely associations between an allowed term for one attribute and a term allowed for another attribute. In the EXPRESS system we have devised a cross-attribute relationship which reflects this association. A typical example of this:

- *hydrogen fluoride* is an allowed term for the attribute *contents*
- *wood pests* is an allowed term for the attribute *range of application*
- *insecticide* is an allowed term for the attribute *product group*

Since the three terms *hydrogen fluoride, wood pests* and *insecticide* are members of different facets in the thesaurus (*contents, range of application*, and *product group* respectively), there would normally be no link between them, as hierarchization takes place solely within the vocabulary of a single facet. However, a product which contains hydrogen fluoride (a poison) is likely to be effective against wood pests, and is likely to be an insecticide. It is therefore compatible with domain knowledge to code a link between the terms indicating just this (see Fig. 3). While investigating the vocabulary in the thesaurus with a view to establishing this relationship between hitherto unlinked terms, it became apparent that in most cases a prognosis could be made as to the direction in which a query would be influenced by changing to the related term. In the example above, switching to the attribute *range of application* and searching for the term *wood pests* will be likely to produce a larger response set than *hydrogen fluoride* in *contents* (since other products which could be used to combat wood pests contain other pesticides). The cross-attribute relationship is therefore always directed (the first argument is assumed to be the more specific term), and thus can be used by the broadening/narrowing algorithms to suggest terms for search reformulation.

Thus thesaurus-based help for a typical search situation is possible: a user specifies a value in a specific attribute as a search parameter, and the expert intermediary informs him or her that a term in a different field would be also/more likely to lead to success, while still retaining the essential sought-for characteristics.

```
facet('insecticide', 'product group').
facet('wood pests', 'range of application').
facet('hydrogen fluoride', 'contents').

cross–attribute('hydrogen fluoride', 'wood pests').
cross–attribute('hydrogen fluoride', 'insecticide').
```

Figure 3 : Excerpt from the EXPRESS thesaurus

Domain-specific associative relationships

The terms within each facet are interrelated by means of the standard thesaurus relationships (generic, partitive, equivalence, associative). If the semantics of the documented domain make it seem expedient, a domain-specific differentiation of the associative relationship can, of course, also be incorporated. Examples for such relationships included in the prototype EXPRESS thesaurus are the 'can-be-made-of' relationship and the 'can-be-treated-with' relationship. The reasoning for the inclusion of these domain-specific associative relationships was pragmatic, the assumption being that a differentiation would allow operationalization for the broadening and

narrowing support functions. For example, if a user is searching for a product with which a *compost fence* can be treated, he or she would probably retrieve more hits using the related term *wood with earth contact*. These two terms are related by the 'can-be-made-of' relationship, which is always consulted by the broadening algorithm.

The embedment of the thesaurus in the EXPRESS system

Figure 4 shows which of the relationships contained in the thesaurus knowledge base are mainly exploited in order to support the three main thesaurus-based functions offered by EXPRESS, i.e. initial query formulation, query reformulation, and browsing. Browsing independently of a given query can, of course, involve all relationships, and can take place concurrently with (re-)formulation. The check-value mechanism otherwise associated with initial formulation can also be invoked during the reformulation process, thus the initial formulation box is contained in the reformulation box.

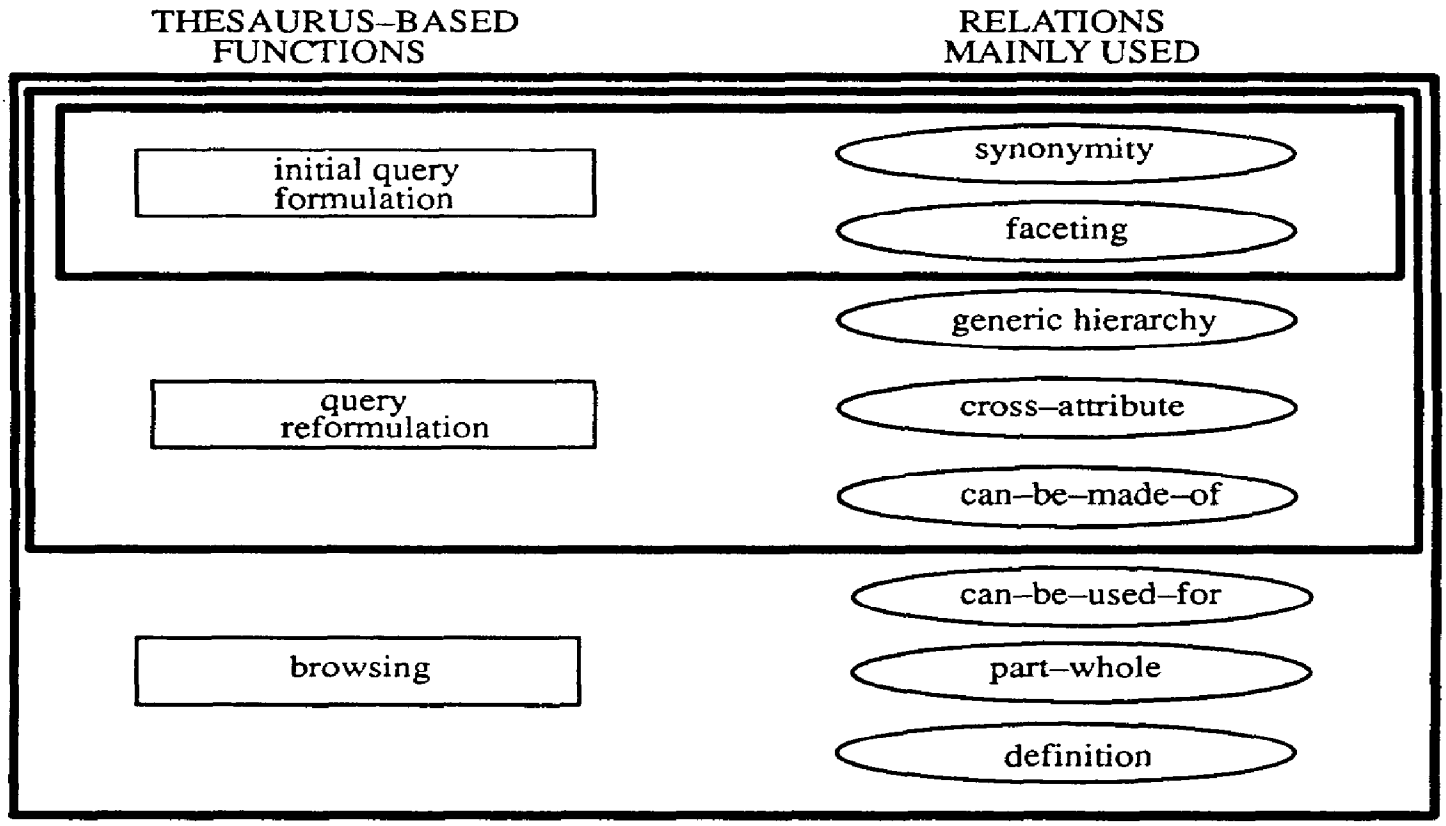

Figure 4 : Correlation between support functions and thesaurus relations used

The thesaurus knowledge base currently contains 347 unique terms, which are interlinked using the eight types of relationships shown in the above diagram. Of these, the generic and partitive relationships are partially defined by means of Prolog rules, i.e. the general *broader-term* relationship is defined recursively as the transitive closure of the explicit, one-step *broader-term1* relationship, and narrower terms are derived by inverting the *broader-term* relationship. Thus, the entire subterm/superterm tree can be retrieved recursively from any given

starting point. In indexing the products in the database, the principle of assigning the most specific (narrowest) descriptor was strictly adhered to. This, together with the recursive expansion, are the basis for the feature that an EXPRESS search query also retrieves all products indexed with the generic narrower terms of the specified search term. When translating a QBE query into SQL, all narrower terms of controlled input terms are retrieved and ORed with the original term. This is an important prerequisite for the functioning of the narrowing algorithm described in the next chapter.

To date the database contains descriptions of 94 wood treatment products. The addition of many more products is planned, whereas the thesaurus has probably achieved its saturation point as far as further growth is concerned. The fact that almost no additional terms were added to the thesaurus during the indexing of the last ca. 20 products tends to confirm that the terminological description of the domain is now comprehensive enough to encompass many additional products.

5 Retrieval tactics and strategies

A common feature of retrieval tactics and strategies is that they lack a generally-accepted definition. Although several definitions (Bates, 1987; Fidel, 1985; Linden, 1987) share a view of strategy as the overall plan of an entire search, they differ essentially in what is regarded as the constituent components (e.g. pre-search interview, selection of hosts and databases, search term selection, reformulation). At least two different basic concepts of strategy are worth mentioning. On one level, each search executed in an online database is conceived of as a search strategy. On another level, there are three strategies for conducting a search (Armstrong & Large, 1988) which are generally recognized, but are not homogeneous with regard to how much they encompass: citation pearl growing, block building, most specific term (facet) first.

We assume strategy to be a broader concept than tactics. A retrieval strategy is thus a combination of tactics, usually performed in more than one search step. Control structures as well as dependencies among query components are characteristic features of a strategy. Moreover, a strategy is characterized by a clearly recognizable direction (i. e. narrowing, broadening, focus shifting), which in turn determines the set of applicable tactics.

A tactic usually consists of one of various kinds of elementary manipulations on search terms, including, for example, their connection with Boolean operators. These tactical manipulations could be classified according to the following categories:

1. Lexical manipulations of search terms including, for example, alternative spelling or truncation;

2. Syntactical manipulations involving the use of Boolean and adjacency operators or alternative spacing;

3. Semantic manipulations using broader, narrower or related thesaurus terms or classification codes; and

4. Functional manipulations on search queries. These include narrowing or broadening of a range of measured values, or a switch to a different attribute in case of functional dependencies

between attributes. The latter plays an important role especially in factual database structures (see previous section on thesaurus).

In our experimental prototype we concentrate on semantic and functional tactics to support the user's reformulation.

Broadening

In order to give advice on how to broaden a query, two main steps have to be performed. First the most appropriate term to broaden has to be detected, whereas there may be more than one. Then a suitable broader term has to be deduced using the various relationships in the thesaurus. In the following, the algorithm is described in more detail:

A) Finding the most appropriate term to broaden:

Examine individual terms

> For all the terms with hit rate 0 check if they are allowed terms using the check-value function (see Chapter 2). If they are not, give the appropriate hints and stop. Otherwise search for all terms with a hit rate less than a given threshold[1] and broaden them (→ B).
>
> If all individual hit rates exceed the threshold continue with the next step.

Examine binary combinations of terms

> For all those pairs which have a hit rate below a given threshold perform the appropriate of the following steps:
>
> - If both terms are descriptors compare the individual hit rates and the sum of the hit rates of their binary combinations and broaden the most restrictive term (→ B).
> - If one of the terms is an entry in an uncontrolled and the other one in a controlled attribute then broaden the descriptor (→ B).
> - If both terms are entries in uncontrolled attributes, propose deleting the one with the lower individual hit rate from the query.

B) Proposing a suitable broader term for the term(s) detected in A:

> Consult the thesaurus and select a relationship according to the following preference list: domain-specific before generic before cross-attribute relationships. Propose using the thus-found term instead of the previous one. If no 'broader' term can be found the term might have to be deleted from the query.

There are some implicit decisions within this algorithm which should be made explicit. For example, descriptors are preferred to non-descriptors because meaningful advice can be given only for them, since only descriptors are interrelated in the thesaurus. It is important to examine

1. The threshold can either be defined as a constant or as a variable which depends on the size of the answer set of the query to be broadened.

combinations of terms because it is often the case that only the combined use of terms restricts the hit rate. On the other hand, for reasons of combinatorial explosion, it is unreasonable to take into account larger subsets of terms than pairs.

Narrowing

The narrowing of a query based on generic thesaurus relations is more difficult to perform than broadening, because it involves selecting from among a potentially large number of more specific terms. It is also not as easy to select the appropriate attribute in which to narrow. An analogous application of the principle used for broadening would be: Select those attributes and terms with the highest individual hit rates. But often these specifications are only used to delimit a certain general area (e.g. a product group), whereas the essential characteristics are specified in attributes which are already more selective, but not selective enough. In light of these problems, we have not yet implemented a sufficiently complete narrowing function in EXPRESS. But, on the other hand, we have exploited some internal mechanisms of EXPRESS, namely the indexing with most specific terms and the automatic recursive term expansion described in Chapter 4, in order to achieve an elegant partial solution.

A) Analysis of the result set:

For the controlled attribute fields, collect resulting terms which differ from the input term. (The query mechanism ensures that these are more specific!)

For all these controlled terms, calculate their relative frequency in the entire answer set.

Select attributes with subterms that exceed a certain minimal relative frequency (→ B).

B) Present the selected attributes and terms with respective relative frequencies to the user.

Order attributes by maximal ratio.

Within an attribute order terms by ratio.

This information allows the user to select appropriate narrower terms according to their relative frequency in the response set for the respective attribute. This criterion, which is independent of the total hit rate, can guide the choice of tactics, for example substitution of a broader term by a narrower term (semantic) or dropping of OR-ed terms (syntactical). In case of absence of suitable narrower terms or rejection by the user, the cross-attribute relation can be used for further suggestions.

Query reformulation as described in the previous sections takes place on the tactical level. The narrowing and broadening support mechanisms which are offered in EXPRESS support the user's decisions as to how the query could be reformulated in order to get a better result. This is based on the assumption that the user's goal, i.e. a specific information need, does not change, and that only the means have to be better adapted to the system. As long as broadening and narrowing are not performed automatically by the system, this is not problematic. But there are important aspects in the evolution of queries which are not captured by these basic functions. The user may subsequently try to pursue different goals in order to extract portions of information and

integrate them later on. The general strategy of block building is an example of this kind of search behavior, but we can also imagine domain-dependent procedures which can only be interpreted and explicitly supported if we know what the information is to be used for. This point is of particular interest for highly structured factual databases.

Specific query patterns and plans

As already pointed out, we assume the database to consist of various relations, each containing a number of attributes. Relational database technology already offers the possibility of defining certain views reflecting specific information needs. A view allows us to combine and select pieces of information in a way which is different from the actual representation in the database. Nevertheless, in order to be useful, a view has to serve a certain class of information needs. Specific information needs have to be mapped onto an adequate view by means of instantiating certain attributes with values and requesting the values of some other selected attributes. Following the "query by example" approach, EXPRESS offers easy-to-use query forms in order to specify such a request. We found that the notion of a **"specific query pattern" (SQP)**, defined in terms of a view with a selection of specified and requested attributes, is a useful basis for further examining the role of procedural task knowledge in retrieval from highly structured databases. Figure 5 illustrates the basic structure of such an SQP, where the attributes to be specified are subdivided into those with constant and those with variable values. Of course, the possible choices of attributes are predetermined by the view.

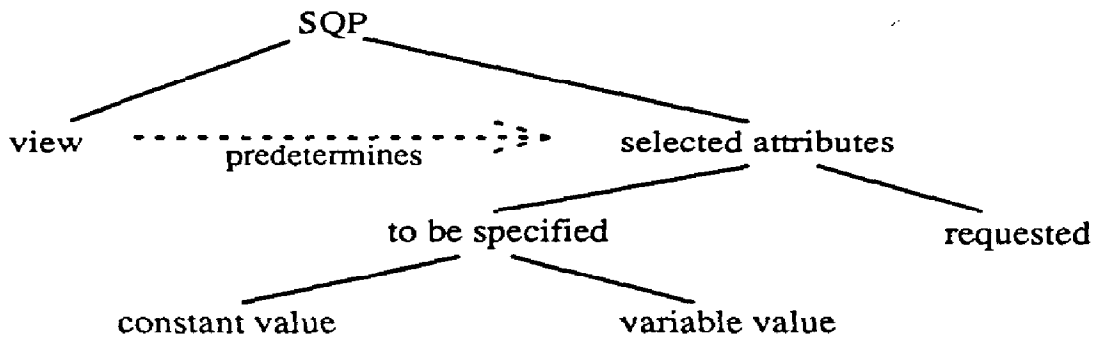

Figure 5 : Structure of a "specific query pattern"

In a small empirical evaluation of protocolled EXPRESS search histories, some SQPs could be identified. The simplest frequent pattern consists of a single requested attribute (*product name* or *effect*) and a single specified attribute (e.g. *range of application* is specified and *product name* is requested in the *product view*). More complex patterns usually evolve as extensions of simpler ones. In the example, the most probable later attribute to be specified in addition to *range of application* is *product group*, followed by *contents*. The specification of additional attributes represents a specific narrowing tactic. Similar tactics based on cross-attribute relationships lead to transitions in which a specified attribute is replaced by another one. But we can also find transitions between SQPs which cannot be interpreted as narrowing or broadening steps. Such transitions agglomerate several SQPs in the form of a task-specific procedural pattern. As yet, SQPs have only been extracted "manually" from dialogue histories. But it should not be too

difficult to use machine-learning techniques in order to extract and generate frequently used SQPs and associated transitions from given protocols automatically. It is one of our current research goals to adapt existing methods for the inductive acquisition of procedural task knowledge (cf. Hoppe & Plötzner, 1989) to the specific requirements of IR.

In order to describe SQPs formally, we suggest the following **notational conventions**: An SQP is defined as a term with two arguments; in the first argument the view is specified and in the second a list of attributes. Within the attribute list, attribute names are identified with variables to be instantiated (in case of requested or user-specified values) or constants (in case of a specified attribute with a constant constraint). Variable names must begin with a capital letter. To-be-specified attributes are marked with the prefix '!'; requested attributes by the prefix '?'. Furthermore, it is useful to distinguish single-valued from list-valued attributes. The latter will be indicated by the suffix '*'. For list-valued attributes which are to be specified, it is important to know their logical connection. If the connection is uniform, e.g. always OR, this is indicated in brackets after the attribute name, otherwise the variable has to be replaced by a pattern containing variable names and logical operators. In the uniform case, the logical operation is interpreted as a prefix operator applied to the list of arguments, so that [NOT OR] is interpreted as a negated disjunction, whereas the specification [OR NOT] would generally not make sense for a list of arguments. Figure 6 shows an example of such a formal description specifying a query in the product view where *product* (the product name) is requested, a disjunction of *ranges of application* may be specified together with a negative specification of a *product group*. Of course, the formal specification of logical operators does not imply that the user has to use the same formal notation. The system may offer the user much easier ways of specifying logical connections.

```
SQP ( view: product-view,
      attributes:    ?product* = Prod,
                     !range-of-application*[OR] = Range,
                     !product-group[NOT] = Group )
```

Figure 6 : An example SQP

Compositions of several SQPs can be regarded as **plans** in the sense of Sacerdoti (1977). The components of such a plan may be partially organized in sequential form, but may also contain order-independent and optional elements. An adequate representation of plans has to account for these possible control structures as well as for hierarchically-nested plan structures and dependencies between parameters (cf. Hoppe, 1988; Schwab, 1989).

The following sequence gives an example of such a plan in the EXPRESS environment: Assume that the user first selects the product view, specifies a product, and requests its contents, purpose(s), and possible range(s) of application. The information on contents, i.e. a set of substances, is then used for specifying the content in the content/effect view in order to obtain a set of possible effects (requested attribute). For a subset of these effects, which are considered to be particularly critical, the user may invert this query pattern by specifying effects and asking for the complete set of critical substances. In the last step, the product view is queried by specifiying

purpose and range as elements of the initially found value sets and specifying the contents as the negated disjunction of critical substances.

Such a plan could be employed to perform the task of assessing the environmental compatibility of some given product and finding less dangerous substitutes. Figure 7 shows how this plan can be represented as a sequence of SQP's with certain parameter constraints expressed in the "where" clause. It is important to note that the final actions are, although highly constrained, not completely determined by the initial query. The concrete instantiation of parameters depends on decisions on the part of the user and has to be handled interactively. Schemata like the one in Figure 7 can be used to monitor the user's task performance, detect potential errors, and suggest further steps or modifications. Suggestions may be presented in the form of optional selection menues.

A plan like "find-substitutes" is clearly domain-specific and should not be confounded with a general retrieval strategy. Nevertheless, general retrieval strategies like block-building or citation pearl growing can be modeled in a similar way.

```
find-substitutes ::=
    sequence-of (   SQP-1 (   view: product-view,
                              attributes:  !product = Prod1,
                                           ?content* = Cont1,
                                           ?purpose* = Purp1,
                                           ?range-of-application* = Rang1 ),
                    SQP-2 (   view: content/effect-view,
                              attributes:  !content*[OR] = Cont2,
                                           ?effect* = Eff2 ),
                    SQP-3 (   view: content/effect-view,
                              attributes:  !effect*[OR] = Eff3,
                                           ?content* = Cont3 ),
                    SQP-4 (   view: product-view,
                              attributes:  !content*[NOT OR] = Cont4,
                                           !purpose = Purp4,
                                           !range-of-application = Rang4,
                                           ?product* = Prod4 ) )
    where       ( Cont2 ⊆ Cont1, Eff3 ⊆ Eff2, Cont4 ⊆ Cont3,
                  Purp4 ∈ Purp1, Rang4 ∈ Rang1 ).
```

Figure 7 : Plan "find-substitutes"

Epilogue: An example dialogue

In order to illustrate the potential benefit of providing interfaces like EXPRESS with a knowledge base of SQPs and plans, let us follow a hypothetical natural language dialogue based on the *find-substitute* plan. We assume that the user has already completed SQP-1 (specifying some product ABC) and the next step SQP-2 (looking at the effects of ABC's contents), which makes it probable that something like *find-substitute* could be intended. So, the system may present the effects (Eff2) and suggest:

.....

System: "Amongst these effects, select those you would like to focus on."

User: (selects some effects)

System: "Would you like to see a complete list of substances which might produce these effects?"

User: Yes.

System: (shows the list)
"Would you like to find other products which do not contain any of these substances and could be used instead of ABC?"

User: Yes.

System: (presents the previous results Rang1 and Purp1)
"Please select the range of application and the purpose for the substitute."

.....

Obviously, this kind of user support is not bound to a natural language interface but could also be provided using graphical presentation and interaction techniques together with "canned text". And of course, the same precautions which apply to any intelligent user interface will have to be taken, e.g. avoid being intrusive, or: do not overload the user with information unless it is really needed in order to perform the task. But considering the current reality of factual Information Retrieval, we find good reasons to try to construct better interfaces for the end user. Providing the interface with more knowledge about terminology, tactics, and strategies might help.

To evaluate and extend the knowledge base in the next version of the EXPRESS system, we plan to empirically identify SQP's by means of automatically analyzing protocols of user queries. This will allow us to assess transition probabilities between tactical moves or shifts of focus, which will then provide a further basis for user support.

Acknowledgements

We would like to thank our colleagues Bernd Kostka and Sylvie Tschumakoff for the implementation of the SunView interface to EXPRESS and other valuable technical support.

6 References

Ammersbach, K. (1986): *Benutzermodelle für Information-Retrieval-Systeme.* Diplomarbeit, Technische Hochschule Darmstadt, Fachbereich Psychologie, Februar 1986.

Ammersbach, K.; Fuhr, N. & Knorz, G. (1988): *Empirisch gestützte Konzeption einer neuen Generation von Werkstoffdatenbanken.* In: Strohl-Goebel, H. (ed.): Deutscher Dokumentartag 1987, Weinheim.

Armstrong, C. J. & Large, J. A. (1988): *Developing search strategies.* In: Armstrong, C. J. et al. (eds.): Manual of online search strategies. Aldershot et al. (Gower), pp. 1-43.

Barthès, C. & Glize, P. (1988): *Planning in an Expert System for Information Retrieval.* In: Proceedings of ACM SIGIR '88, Grenoble, pp. 535-550.

Bates, M. (1987): *How to use Information Search Tactics online.* In: ONLINE, May 1987.

Brajnik, G.; Guida, G. & Tasso, C. (1987): *User Modeling in Intelligent Information Retrieval.* In: Information Processing & Management, Vol. 23, No 4, pp. 59-63.

Brooks, H.M., Daniels, P.J. & Belkin, N.J. (1985): *Problem Descriptions and User Models: Developing an Intelligent Interface for Document Retrieval Systems.* In: Advances in Intelligent Retrieval, Proceedings of Informatics 8, London: Aslib, 1985, pp. 191-214.

Brooks, H.M. (1987): *Expert Systems and Intelligent Information Retrieval.* In: Information Processing & Management, Vol. 23, No 4, pp. 367-382.

Card, S.K., Moran, T.P. & Newell, A. (1983): *The Psychology of Human-Computer Interaction.* Hillsdale, NJ: Erlbaum.

Fidel, R. (1985): *Moves in Online Searching.* In: Online Review 1985, Vol. 9, No 1, pp. 61-74.

Fischer, G. & Nieper-Lemke, H. (1989): *HELGON: Extending the Retrieval by Reformulation Paradigm.* In: Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems, Austin, Texas, April/May 1989, pp. 357-362.

Hayes-Roth, B. & Hayes-Roth, F. (1979): *A Cognitive Model of Planning.* In: Cognitive Science, Vol. 3/4, pp. 275-310.

Hoppe, H.U. (1988): *Task-Oriented Parsing – A Diagnostic Method to be Used by Adaptive Systems.* In: Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems, Washington D.C., May 1988, pp. 241-247.

Hoppe, H.U. & Plötzner, R. (1989): *Inductive Methods for Acquiring Task-Knowledge in Adaptive Systems.* GMD-Tech. Report No. 392, Birlinghoven, FRG.

Kobsa, A. & Wahlster, W. (eds.) (1989): *User Models in Dialogue Systems.* Berlin-Heidelberg-New York: Springer.

Linden, F. (1987): *Wissensgestützte Datenbankauswahl und Recherche.* WBS-Bericht 5/87, TU Berlin, Inst. f. Angewandte Informatik.

Mannes, S. & Kintsch, W. (1989): *Action Planning: Routine Computing Tasks.* In: Proceedings of 10th Conference of the Cognitive Science Societ,. Montreal, Canada, August 17-19, 1988, pp. 97-103.

McAlpine, G. & Ingwersen, P. (1989): *Integrated Information Retrieval in a Knowledge Worker Support System.* In: Proceedings of ACM SIGIR '89, Cambridge, MA, June 25–28, 1989, pp. 48–57.

Monarch, I. & Carbonell, J. (1986): *CoalSORT: A Knowledge-Based Interface to an Information Retrieval System.* Tech. Report, Carnegie-Mellon University, Pittsburg, PA.

Payne, S.J. & Green, T.R.G. (1986): *Task-Action Grammars – a Model of the Mental Representation of Task Languages.* In: Human-Computer Interaction, Vol. 2, pp. 93-133.

Rich, E. (1979): *Building and Exploiting User Models.* PhD Thesis, Computer Science Department, Carnegie-Mellon University, Pittsburg, PA.

Sacerdoti, E. (1974): *A Structure for Plans and Behavior.* Amsterdam: North-Holland.

Schwab, T. (1989): *Methoden zur Dialog- und Benutzermodellierung in adaptiven Computersystemen.* PhD Thesis, Univ. Stuttgart: Inst. f. Informatik.

Shoval, P. (1983): *Knowledge Representation in Consultation Systems for Users of Retrieval Systems.* In: The Application of Mini- and Micro-Computers in Information, Documentation and Libraries. Amsterdam: North Holland, pp. 631-643.

Smith, P.J.; Steven, J.S.; Galdes, D. & Chignell, M.H. (1989): *Knowledge-Based Search Tactics for an Intelligent Intermediary System.* In: ACM Transactions on Information Systems, Vol. 7, No 3, pp. 246-270.

Vickery, A. & Brooks, H.M. (1987): *PLEXUS – the Expert System for Referral.* In: Information Processing & Management, Vol. 23, No 2, pp. 29-117.

Zloof, M.M. (1983): *The Query-by-Example Concept for User-Oriented Business Systems.* In: Sime, M.E. & Coombs, M.J. (1983): Designing for Human-Computer Communication, London: Academic Press, pp. 285-309.