Chapter 8

An Expert Interface for Effective Man-Machine Interaction

Giorgio Brajnik, Giovanni Guida*, and Carlo Tasso**

Abstract. This chapter presents a prototype expert interface (IR-NLI, Information Retrieval Natural Language Interface) to an information retrieval system, developed at the University of Udine in the frame of a broader research effort concerning the topics of cooperative man-machine interaction and expert systems. After a discussion of the novel notion of expert interface, attention is focused on the IR-NLI system for the access to online information services by non-technical users. General specifications, design criteria, and architecture of IR-NLI are presented first. Knowledge representation methods and reasoning mechanisms adopted are then illustrated in detail. In this context, a new mechanism, called task, for representing and using meta-knowledge in rule-based systems is proposed. The internal operation of the system, together with two examples of interaction with the user, are illustrated. The paper concludes with the discussion of some preliminary ideas on how learning capabilities could be introduced in IR-NLI through the task mechanism.

1 Introduction

The study and design of friendly man-machine interfaces has seen remarkable progress in the last decade. Natural language has always been considered a key point in bridging the gap between a casual user and a complex system. In fact, the first difficulty one encounters in approaching an unknown machine and trying to use it, arises from the fact that the machine speaks a language different from that of the user: he cannot talk to it in his native language and he cannot expect to engage in a friendly dialogue. Much effort has been devoted to the study of natural language communication between man and machine, and presently a number of implemented systems exist that can demonstrate enough coverage and robustness in limited subject domains, and can be usefully adopted in real applications. Nevertheless, current interfaces still lack domain competence and their behaviour - although quite natural and friendly - is not cooperative enough to support an effective interaction.

* Also with: Artificial Intelligence Project, Politecnico di Milano, Milano, Italy.
** Also with: CISM - International Center for Mechanical Sciences, Udine, Italy.

A brief look at the recent history of the role of natural language processing in the design of man-machine systems can help in analyzing the general trends of this research area. Tracing back to the early seventies, some milestones can be identified in the development of the concept of man-machine interface. We can sum up this evolution in five major steps:

Understanding Imperative Natural Language Sentences
The main goal considered in this first phase is that of enabling a system to understand separate queries or commands expressed in natural language: no shared world knowledge between the system and the user is provided. Several generations of natural language interfaces up to the beginning of the seventies conform to this paradigm [GREE63, LIND63].

Managing Simple Dialogues
The issue of carrying on a (simple) dialogue with the user arises quite naturally from the exigency of overcoming critical situations of misunderstanding. The notion of clarification dialogue to support the system in the comprehension of difficult (imperative) sentences is a good example of this trend [WOOD72, CODD74].

Having a Graceful Behavior
Managing a graceful interaction between man and machine, including most of the features that make a natural conversation easy and pleasant emerges as a necessary capability of a dialogue system. Understanding fragmentary and broken text, dealing with grammatically incorrect sentences, getting confirmation of the correct understanding of an utterance, managing ellipsis and anaphora are only a few of the issues involved in graceful man-machine interaction [HEND78c, HAYE79].

Taking Pragmatics into Account
Several of the above issues point to the need of having complex pragmatic models of communication, beyond syntax and semantics. Knowing how the actors of a dialogue behave, how they manage interpersonal plans, how they use language as a medium to achieve their goals, becomes now a crucial point in the design of effective interfaces [ALLE83, WILE83]. Linguistic knowledge is not enough to master such complex situations: a lot of world knowledge is needed.

Being Cooperative and Supportive
Taking active part into a conversation is not limited, however, to mastering language and world knowledge: an intelligent interface should be able to contribute to the dialogue with its own reasoning capabilities. A natural man-machine interface should be cooperative and supportive for the user and actually help him in the solution of his problems [COHE82, KAPL83, REIC84, REIT83].

This short history denotes a clear trend to move the focus of attention from merely linguistic phenomena towards a more comprehensive understanding of the nature of man-machine interaction, taking into account phenomena that lie behind the linguistic surface and that are expected to play a crucial role in the design of smarter and more effective interfaces. Language is obviously a major problem in man-machine communication, and the use of natural language can surely contribute to bridge the gap between machines and non-technical users. However, language is neither the only impediment nor the main one to an easy and effective use of complex artificial systems. Concepts, models, world knowledge, mental paradigms, intentions, plans, needs, roles which are behind linguistic utterances constitute the real gap between man and machine.

Very often a machine is easily usable only by a person who exactly knows which tasks the machine can perform, how it can be operated, and how it works. But non-technical users are generally not acquainted with all these matters; they only have a rough idea on how to use the machine they are willing to interact with, and some expectations on its capabilities. Moreover, the machine does not care about the user: it does not embody knowledge about possible user profiles, and it does not take into account how its performance can be interpreted by the interacting person and how it can affect his behaviour.

Therefore, the deepest gap between man and machine lays, first of all, at the conceptual level; diversity of language reflects diversity of concepts, and trying to bridge the linguistic gap between man and machine requires, as a prerequisite, a huge amount of shared world knowledge. Only after the conceptual gap is bridged, it is appropriate to address the problem from the linguistic point of view: a neat conceptual context can be an adequate basis to deal with naturalness, robustness, and cooperativeness of discourse.

In this chapter we propose an approach to the design of cooperative man-machine interfaces which explicitly takes into account the conceptual point of view, both to support cooperation and, also, to improve the level of linguistic communication. In order to identify the basic capabilities of such an interface, let us consider a paradigmatic situation where a person (the user), facing a problem in a given subject domain, realizes that some artificial system (the target system) can serve his purpose, and, therefore, decides to use it. But, he does not know enough about how to operate the target system correctly and how to utilize it effectively.

It is straightforward to recognize that the best possible interface - both natural and cooperative - between the user and the target system is, in such a situation, a human intermediary. His expertise covers target system capabilities, use, and operation; he would know about application domains where the target system can be effectively used; he also is knowledgeable about needs, expectations, and mental models of potential users; at last, he fluently speaks the same (natural) language of the user. Such an intermediary can perform, therefore, as an intelligent and active problem solving assistant who can support a wide class of users in the correct and effective use of the target system.

The role of the intermediary between the user and the machine includes three main capabilities:

- assisting the user in the correct and effective use of the target system;
- taking active part in the problem solving process in which the user is engaged;
- possibly training the user in the operation of the target system, gradually making him capable of using it by himself.

In order to perform the job outlined above, the intermediary must have a huge amount of knowledge that can be organized in four classes:

Knowledge About the User:
- problem solving situations where the target system could be used (from the viewpoint of the user);
- user view on the problem domain, including world models and reasoning mechanisms;
- specific exigencies the user wants to satisfy by means of the target system;
- attitude of the user towards the intermediary (cooperative, uncooperative, misleading, suspicious, etc.);
- interest of the user in knowing more about the target system and acquiring the capability of using it directly;
- interpersonal communication model adopted by the user, including language use and terminology.

Knowledge About the Target System:
- functions and limitations;
- operation (how to run it appropriately), including knowledge about the command language;
- general architecture and internal operation;
- problem solving situations where it can be utilized;
- specific use in the solution of classes of problems.

Knowledge About the Problem Domain:
- general knowledge (taxonomic, descriptive, theoretical) on the subject domain;
- general reasoning strategies;
- domain specific (heuristic, experience-based) reasoning mechanisms;
- classes of usual problems in the subject domain.

Knowledge About the Intermediary's Job:
- models of intermediary's professional activity (goals, resources, strategies, tactics, etc.);
- specific models of interpersonal communication (interviewing, convincing, etc.);
- specific (heuristic, experience-based) skills of the intermediary's job.

So far, we have analyzed the main features of a person, a skilled intermediary professional, who is supposed to behave as a living interface to a target system. Our proposal consists in putting the intermediary's competence into an artificial system, which is expert in the specific task of interfacing non-technical users to a complex system. We call this kind of interface an *expert interface*.

The purpose of this paper is to illustrate the research activity performed in the design and experimentation of the *IR-NLI (Information Retrieval Natural Language Interface)* system for the access of non-technical users to online information services. IR-NLI is an expert interface modeled according to the general criteria outlined above, which includes both linguistic coverage and expert competence, and can offer a cooperative and supportive interaction environment. This project has been developed at the University of Udine [GUID82, GUID83 a, GUID83 b] with the aim of experimenting with new techniques for the design of cooperative interfaces and, also with some new ideas concerning the architecture of expert systems [GUID84]. IR-NLI is currently running on VAX-11/780 (it is written in Franz Lisp under the UNIX operating system), and has been tested in the domain of bibliographic information retrieval in a subfield of computer science.

The issue of interfacing online information retrieval systems with user-friendly and cooperative front-ends is not new in the literature. CITE [DOSZ79] is an English language interface for querying an online bibliographic system. Its capabilities include identification of search terms, combinatorial searching, ranked output, relevance feedback, and automatic query modification. CITE does not utilize any of the usual AI techniques for automatic reasoning, and it is therefore unable to include in the search strategy terms which are not supplied by the user in the initial request. Therefore, CITE is mainly addressed to experienced users, who are well acquainted with the most appropriate terminology.

The search statement generator developed by Pollitt [POLL81, POLL82] is an interface to a medical data base, which can be utilized to generate appropriate search commands with a suitable terminology in the domain of cancer therapy. Although sharing with IR-NLI the goal of enabling untrained end users to easily access and search an online database, Pollitt's system is mainly concerned with the problem of constructing and evaluating a non-typing interface, where a touch screen is utilized by the user in order to select the most appropriate concepts, terms, and commands from frames displayed on the screen.

The CONIT system [MARC81 a] is a specialized interface designed to assist a user in directly interacting with several different information retrieval systems. CONIT makes available to the user a generalized query language, and can translate commands of this language into the appropriate commands needed to access whatever specific system is being interrogated. Moreover, CONIT can teach the user the most correct way of utilizing the different commands, and can suggest possible search techniques to be employed. The user is then responsible to decide how to apply these techniques by using the appropriate commands. CONIT does not address the issue of supporting the user in the formulation of the search strategy at a conceptual level, but only provides an instruction aid.

A different, more ambitious point of view is taken in the design of EXPERT [MARC81 b]. Based on the previous experience developed with CONIT, an experimental computerized intermediary system has been developed, which is able to simulate a human expert in the task of assisting inexperienced users of bibliographic retrieval systems. EXPERT interacts with the user through a menu-selection/fill-in-the-blanks dialog which assists the user in data base selection (relying on statistical measures) and supports search formulation and reformulation (this last obtained through relevance feedback techniques). Moreover, EXPERT performs an automatic translation of the search formulation into keyword/stem boolean search commands. EXPERT is implemented as a production system controlled by a forward chaining mechanism. With respect to CONIT, it offers substantially improved capabilities, but it still does not take into account any domain-specific knowledge. Therefore, it can effectively support only users fully acquainted with the subject domain of the search.

All the projects outlined above, although sharing with IR-NLI the same application domain, substantially differ from it in scope. The problems they face are mainly at the linguistic level, and only partially deal with such basic tasks as the analysis of the user's information needs and the design of an appropriate search strategy. Moreover, several crucial issues of conceptual nature, including cooperative behaviour end expert problem solving, are not explicitly addressed.

This chapter is organized as follows. In Sect. 2 a brief introduction to online information retrieval is presented. Section 3 is devoted to the analysis of the specifications of IR-NLI and to the illustration of its general architecture. Section 4 focuses on knowledge representation, and Sect. 5 on the internal operation of the system. In Sect. 6 two examples of interaction with IR-NLI are presented. Section 7 discusses some preliminary ideas on how learning capabilities can be introduced into IR-NLI, and Sect. 8 concludes the chapter.

2 The Information Retrieval Field

2.1 General Issues

In this section we illustrate the basic features of online information retrieval [LANC79, SALT83], the subject domain of the IR-NLI expert interface.

Online services allow interested users to solve information problems by selecting and retrieving relevant documents stored in very large bibliographic or factual data bases, concerning several fields of science, technology, economics, and humanities. *Information retrieval systems* allow access to stored data through a formal query language, which is used for selecting the desired information through a sequence of commands (such as SELECT, FIND, DISPLAY, logical combination of conditions, etc.). Generally, end-users are unwilling (or unable) to interrogate these large files directly, and they rely on the assistance of a specialized information professional, the *intermediary,* who knows how to select appropriate data bases, how to design good search strategies for the retrieval of the desired information, and how to implement them through a suitable formal query language.

In order to clearly point out the peculiarities of this application domain, it is useful to contrast information retrieval systems to data base management systems. We consider two main points.

First, the information stored in a data base has a well defined logical structure, which supports the access to data and helps the user in finding desired information items. In contrast, in an information retrieval system, the stored records (which contain, for example, title, authors, abstract, key words, etc. of a technical paper) identify the content of the documents only in a partial and unstructured way. This implies that retrieval of desired information is much more difficult, since it relies on several loosely defined factors, such as domain-specific knowledge, knowledge about indexing criteria, availability of updated and complete searching referral aids, working experience on the particular database, etc. This first feature is captured by two classical parameters used to measure the effectiveness of a retrieval: recall and precision [SALT83].

Second, while a user of a database generally looks for precise information items to be extracted from the stored files, a user of an online service desires to get information on a given topic, and this does not allow straightforward and exact identification of the relevant records. What documents to extract in order to match the user's request is usually a non-trivial problem that has to be solved by the intermediary.

2.2 *The Actors and Their Role*

Let us continue with the illustration of the scenario of information retrieval by introducing the three actors of a usual search session:

- *the end user:* the person who has some information need that can be satisfied by means of an online service;

- *the intermediary:* the professional who is able to understand and analyze the information needs of the end user and to effectively access the data base in order to retrieve the relevant documents;

- *the data base:* a file of bibliographic (or factual) records which contain references to the literature of a specific subject domain. Generally, a given topic can be dealt with in several data bases, which may differ in the general methodology utilized for indexing and storing documents. Information about the technical peculiarities and the content of a data base (including the criteria employed for collecting relevant information, the attention given to theoretical or practical aspects of the subject, the structure of the records, the list of key-words utilized for indexing the documents, the use of special classification schemata, etc.) is supplied by *searching referral aids*, such as reference manuals, thesauri, directories, subject headings, and classification plans.

The interaction between these three actors usually follows a well defined and established procedure, which is structured in the following four phases [MEAD81]:

1. *Presearch Interview*
   It is aimed at precisely clarifying the content and the objectives of the end user's information needs. In this phase the intermediary performs the *conceptualization* of the information problem, i. e. he identifies all concepts present in the end user's request and he analyzes the logical relations existing among them.

2. *Data Base Selection and Strategy Design*
   On the basis of the information gathered during the presearch interview and with the help of searching referral aids, the intermediary chooses the most suitable data base(s) to be interrogated and devises a search strategy which matches end-user goals and needs. By the term *search strategy* we refer to the formal query program (a sequence of statements written in the specific query language of the chosen data base), which can be submitted to the information retrieval system in order to select the relevant documents.

3. *Execution*
   Once a tentative search strategy is available, the intermediary submits it to the information retrieval system, collects the results of its execution, and evaluates them in order to refine, if necessary, the search strategy. Sometimes, phases 2 and 3 are repeated more than once to produce a final version of the search strategy which better fits the end user's needs.

4. *Result Evaluation*
   The output of the search is evaluated by the end user, who may decide to change his requirements (add or substitute some terms, adjust objectives, etc.) or to propose some modification to the search strategy adopted.

From this illustration of the main phases of a search session, it can be noticed that one of the most critical tasks to be performed by the intermediary is the design of a suitable search strategy. In the next section, we illustrate this phase in greater detail, focusing, more specifically, on some of the techniques currently utilized by professional intermediaries.

2.3 Approaches and Tactics for Strategy Design

The term *approach* denotes the abstract way of facing a search problem, reasoning on it, analyzing its facets, and devising a general mode of operation to access the desired information. More precisely, an approach defines the order and the way of combining (through logical operators) the relevant concepts in the design of a search strategy. Five main approaches have been proposed in the literature [MEAD81]:

Most Specific First (MSF)
In this case attention is first focused, in a multifaceted end user's request, on the concept which is the most specific in the query. If, after the search, the number of the retrieved documents is sufficiently low, the search is ended. Otherwise, the remaining concepts are taken into account and the search is repeated. The MSF approach is useful when dealing with a query from which it is possible to extract a concept which is much more specific than the others. In this case, the MSF approach yields the results through very few access operations.

Lowest Posting First (LPF)
This approach is similar to MSF: the concept with the lowest posting count (the number of documents in the data base which are related to a given concept) is searched first; then the results are evaluated and, if necessary, the next lowest posting concept is searched again until a reasonable set of documents is retrieved. Usually, the LPF approach is chosen when the query contains a concept whose posting count is much lower than the others.

Building Block (BBL)
In this approach the query is broken into several concepts or groups of concepts - the facets - and each of them is searched independently. The analysis of each concept (search for related terms, synonyms, possible abbreviations, etc.) is carried out off-line, and only after its completion the information retrieval system is used. The BBL approach involves the lowest interaction degree with the data base system. It is used when a very exhaustive search is desired.

Citation Pearl Growing (CPG)
This is the most empirical and interactive approach: starting from a concept - usually the best known one - a first search is performed; the evaluation of the results so far gathered will then provide further citations and terms which can be used in a subsequent phase of the search. In most cases this try-and-evaluate cycle is repeated a few times until enough information has been collected which eventually leads to the selection of another approach. The CPG approach is mainly used when the user is poorly acquainted with the specific matter, and he does not know the precise terms to be used.

Successive Fractions (SFR)
In this approach, starting from a concept or a limitation specified by the user on the desired documents (e.g., the date, the language, etc.), a search is performed;

another condition (e.g., a concept) is then imposed on the results so far retrieved and a new search is executed. This is repeated until either there are no more conditions to be taken into account or the obtained results are satisfactory.

During the execution of an approach a fundamental activity performed on the facets of a query is *concept analysis*. In this activity, a basic role is played by tactics, which were first defined by Bates [BATE79] as a move made to further a search. From this point of view, a tactic can be understood as a fixed sequence of actions devoted to carry out the analysis.

In our framework the term tactic has been assigned a more specific meaning: a *tactic* is an elementary operation, a single step or action to be performed in order to implement an approach. We can therefore further characterize the concept of search strategy in the following way: a search strategy is the result of the execution of an approach through the application of appropriate tactics. A list of the main tactics usually utilized during concept analysis (BATE79, LANC79] is reported below:

Specify: the search is to be carried out on terms which are more specific than the ones used so far. This tactic can be used when high precision is needed and the current formulation of the query is likely to produce a huge amount of results.

Generalize: the search has to be extended by moving up in the term hierarchy to more general terms and by including several combinations of them in the search formulation. It can be used when the specificity of the current search formulation is too high with respect to the search objectives, and it is likely to produce too few and/or too specific items. It is the opposite of specify.

Exhaust: the search formulation has to include (ANDed together) most of (or all) the terms of the query. This tactic leads to the retrieval of a restricted set of documents, perhaps missing some important ones due to overly stringent conditions.

Reduce: the search formulation has to be relaxed by eliminating one or more of the terms present in it. Reduce is the opposite of exhaust.

Parallel: the search formulation has to be enlarged by including (ORed together) synonyms or related terms. This tactic can be used when the elements in the query are too specific and precise.

Pinpoint: the precision of the search formulation has to be increased by reducing the number of parallel terms. This considerably reduces the degree of recall.

Super: moving upward in the term hierarchy and focusing on a broader term, discarding the original one.

Sub: moving downward in the term hierarchy and focusing on a more specific term, discarding the original one.

Relate: moving at the same hierarchical level towards related terms, and substituting the original term with the new ones (ORed together).

Sibling: including in the search formulation some related terms ORed together with the currently analyzed term. This tactic increases recall, but, at the same time, it is much more reliable than the relate tactic, as it still keeps in consideration the original term.

Superordinate: including in the search formulation some related terms ORed together with the currently analyzed term, and ANDing the result with a broader term. This tactic broadens the search to related terms, still preserving proximity with the original one (because of the broader term ANDed with the related ones). Superordinate can be used when a high recall is desired, without paying the price of a crude broadening.

Subordinate: including in the search formulation some narrower term ORed together, and ANDing the result with the original term. Subordinate is the opposite of superordinate, as it involves a specification over the term rather than a generalization. It can be used when a higher specificity is needed, without the risk of an arbitrary specification.

Rearrange: if a term contains more than one word, reverse or rearrange the words in any reasonable order.

Respell: searching under different spelling variants of the terms.

Respace: searching under different spacing variants of hyphenated words.

Fix: searching under different prefix, subfix, or infix variants.

3 System Architecture and Specifications

3.1 Overview

The general architecture of IR-NLI is shown in Fig. 1. IR-NLI comprises three main modules, devoted to cover the three basic competence areas of the interface, namely: natural language understanding and dialogue, modeling of the intermediary's activity, and generation of the search strategy in the appropriate command language of the target information retrieval system.

The *understanding and dialogue module* is devoted to perform activities of linguistic nature. First, it translates the natural language user's request into a formal *problem internal representation*. Second, it manages (under the control of the reasoning module) a dialogue with the user devoted to expand the problem internal representation with new information. The understanding and dialogue module utilizes for its operation a *vocabulary* and a base of *linguistic knowledge*.

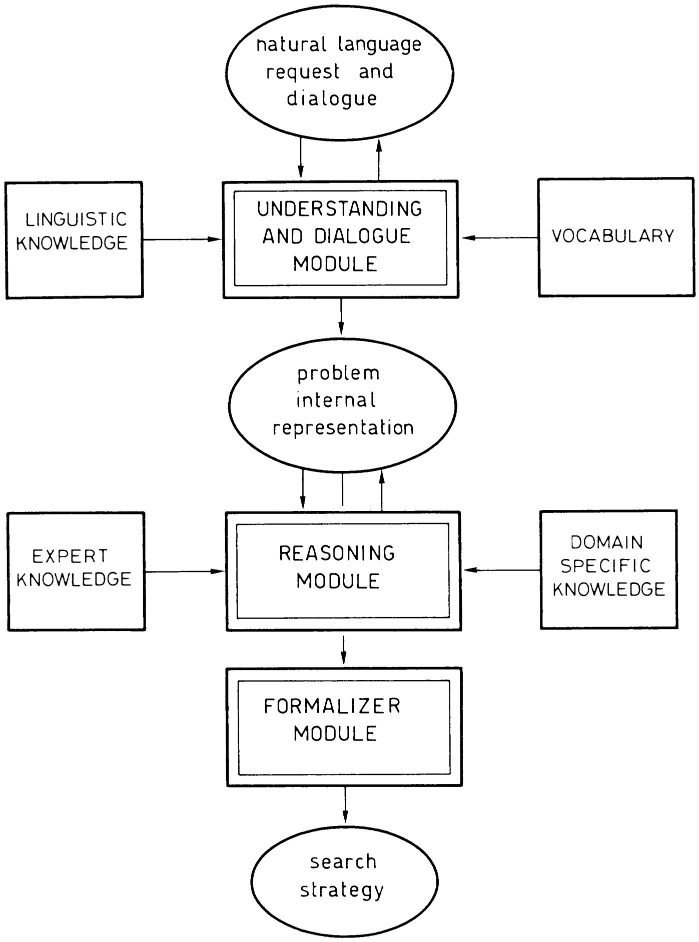
natural language request and dialogue
LINGUISTIC KNOWLEDGE
UNDERSTANDING AND DIALOGUE MODULE
VOCABULARY
problem internal representation
EXPERT KNOWLEDGE
REASONING MODULE
DOMAIN SPECIFIC KNOWLEDGE
FORMALIZER MODULE
search strategy

Fig. 1. Overall architecture of IR-NLI

The *reasoning module* is devoted to model the intermediary's activity: it devises the top-level choices concerning the cooperative operation of the interface and controls their execution. To perform this activity it uses a base of *expert knowledge,* which concerns the evaluation of user's request, the management of the presearch interview, the expansion of the problem internal representation, the selection of a suitable approach, the execution of appropriate tactics, the activation of the dialogue with the user, and the generation of a suitable search strategy. The world knowledge (mostly of terminologic nature) necessary to these tasks is contained in a base of *domain specific knowledge.*

The *formalizer module* is activated after the reasoning module has completed its activity, and constructs from the fully expanded problem internal representation the final search strategy to be executed for accessing the target online data base. It is conceived as a parametric translator, capable of producing search strategies in several languages for accessing actual online services, such as SDC ORBIT, Euronet DIANE, and Lockheed DIALOG.

The brief illustration of the overall system architecture shows that the reasoning module constitutes the kernel of IR-NLI, as it is devoted to provide the system with the cooperative behavior typical of an intelligent information retrieval assistant. Therefore, we focus in the following sections only on this part of the system, which embodies the conceptual capabilities that make IR-NLI an example of an expert interface as defined in Sect. 1.

The design of the understanding and dialogue module closely follows the goal-oriented approach developed by the authors and tested in the construction of the NLI system for the enquiry in natural language of a relational data base [GUID82]. The formalizer module can be easily implemented with traditional syntax-directed translation techniques.

3.2 A Rule-Based Architecture for the Reasoning Module

We focus in this section on the basic requirements of IR-NLI and we derive a first set of general technical specification for the reasoning module. The analysis carried out in Sect. 1 on the types of knowledge involved in the intermediary's job, clearly suggests that the rule-based system technology [WATE78, HAYE83] can offer a viable approach to the design of the reasoning module. In fact, several reasons support this choice.

First, a variety of different knowledge types are involved in the subject domain of intelligent information retrieval, including:

- precise classification information about the domain of the search, elicited from thesauri and searching referral aids and stored in the domain-specific knowledge base;
- structured knowledge extracted from technical literature and practical experience about approaches, tactics, and their use in intelligent information retrieval;
- uncertain, judgmental, and often even conflicting knowledge about how to model the intermediary's skill, derived from the analysis of his operation in practical situations;
- incomplete and unreliable data contained in the user's request (often crucial terms or references are missing, while unimportant or misleading concepts are stressed).

Furthermore, the reasoning paradigms of the intermediary are typically data-driven, as they highly depend on the several specific situations which can occur during the search. Therefore, it would not be appropriate to implement them in a fixed algorithm, which should explicitly consider all possible cases.

Moreover, knowledge about the intermediary's job seems to be more easily represented in a declarative way, rather than through fixed procedures. In fact, knowledge necessary to the effective operation of the system is not likely to be available at once before implementation, but it can be acquired, refined, and tested incrementally during system experimentation and tuning.

At last, the search space is estimated to be very large and easily factorable.

3.3 Technical Specifications

The task domain of intelligent information retrieval can hardly be classified in any of the categories of rule-based systems applications defined in [HAYE83]. In fact, it shows several features of different categories, including:

- *Interpretation:* the user's request can, generally, not be assumed to be a faithful description of his real information needs, since it is often incomplete, ambiguous, uncertain, and misleading. More appropriately, the initial user's request can be considered a set of raw data which have to be carefully interpreted in order to capture their deep meaning.
- *Planning:* carrying out a successful dialogue with the user and performing a correct analysis of his request require the basic capability of planning (and dynamic replanning) the choices and actions to be taken.
- *Design:* the construction of a suitable search strategy satisfying given objectives and constraints is a typical design problem, which requires the capability of considering large classes of possible solutions and of taking decisions according to appropriate evaluation criteria.

Sharing features of several application categories constitutes a first impediment to the adoption of traditional rule-based system architectures and suggests that a new paradigm capable of supporting cooperation among interpretation, planning, and design has to be looked for.

Moreover, a first implementation of IR-NLI developed in the past [GUID83 b] has shown the inadequacy of a naive approach, based on usual rule-based system techniques, and it has outlined several features that denote the need for new tools capable of fitting the specific peculiarities of an expert interface.

The following technical features can be mentioned:

- The search space is very large, and it shows a tree-like structure comprising loosely connected knowledge islands, i.e. well structured chunks of rules corresponding to the different approaches and tactics of online information retrieval. Therefore, high directionality and specificity of search (guessing) is needed.

- Moreover, in order to ensure a correct recovery in case of a failure during the search, intelligent backtracking capability to the appropriate branching point is necessary. In fact, some kinds of failure may suggest to move to different knowledge islands without trying all alternatives offered by the current one.
- Furthermore, the knowledge island structure of the rule base demands a long-term planning capability which can avoid a fragmentary operation without look-ahead.
- As knowledge about information retrieval naturally comprises a hierarchy of abstraction levels (e.g., rules about approach selection are at a meta-level with respect to rules about tactics to be used for approach implementation), adoption of a mechanism for rule interpretation is suggested which can take advantage of this knowledge structure for improving efficiency of both matching and conflict resolution.
- At last, as the ability of the intermediary is largely based on analogical reasoning, mechanisms of learning from experience should be provided, which enable the reasoning module to refine its skill with operation [CARB83b].

Several of the issues outlined above have major implications for the design of the architecture of the rule-based system for the reasoning module and, more specifically, for the organization and use of meta-knowledge. Therefore, the need arises for a knowledge organization which can fit the many different knowledge types involved in the information retrieval task, and for a mechanism capable of exploiting meta-knowledge in an effective manner.

4 Knowledge Representation

4.1 Domain Specific Knowledge

The *domain specific knowledge base (DKB)* contains knowledge about the subject domain covered by the target online data base to which IR-NLI is connected. The DKB embodies two kinds of knowledge [DEFU84]:

- *linguistic knowledge,* concerning how a concept is currently referred to in the data base through an appropriate linguistic item specific of the subject domain;
- *semantic knowledge,* which expresses classification information and cross-reference relations among terms.

The internal structure of the DKB has been modeled according to the organization of traditional searching referral aids (in particular thesauri and subject classifications). It is constituted by a labelled directed network, in which nodes represent concepts and directed arcs represent relations between concepts. Each node contains:

- a *term,* i.e. the appropriate linguistic item used in the specific domain to denote a concept;
- a boolean flag (Y, N), which indicates whether the term is a *controlled term (CT),* i.e., whether it belongs to a controlled vocabulary (a collection of standard terms used for document indexing);
- the *posting count (PC),* i.e., the number of items of the data base in which the term appears;
- an integer value, called *generality level (GL),* which represents the degree of specificity of the term in a hierarchical subject classification.

Arcs denote the usual cross-reference relationships utilized for structuring thesauri. These include:

- *broader term (BT),*
- *narrower term (NT),*
- *related term (RT),* and
- *used for (UF).*

The structure of the DKB has been designed in such a way as to be easily constructed from available searching referral aids and online thesauri, possibly in a partially automatic way through appropriate conversion programs.

4.2 Expert Knowledge: Rule Structure and Organization

In addition to the domain specific knowledge, the reasoning module utilizes in its operation an *expert knowledge base (EKB),* which contains knowledge about the intermediary's skill and expertise. Expert knowledge takes the form of *production rules.*

The *left-hand-side (LHS)* of a rule is constituted by a sequence of logical *conditions* defined over the problem internal representation, that are assumed to be ANDed together. We say that the LHS of a rule is *satisfied* if all conditions in it evaluate to true. The *right-hand-side (RHS)* of a rule can contain:

- *actions* to be executed on the problem internal representation (possibly including interaction with the user or with the DKB), and
- *directives,* i.e. operations which will affect the control flow of the system.

Actions and directives in the RHS of a rule are assumed to be performed, when the rule is executed, in a strictly sequential manner.

Let us now take a closer look at the internal structure of a rule. A rule takes the form of a list of descriptor-value pairs: the *descriptor* is an identifier, and the *value* is either an atomic object (a number, a string, etc.) or a piece of Lisp code. The descriptors currently used for rule representation are:

- ⟨*name*⟩: an identifier (atomic symbol) used for referencing the rule in the EKB;
- ⟨*type*⟩: the type of the rule (domain, matching, or conflict);
- ⟨*weight*⟩: a real value between 0 and 1, which expresses a measure of how reliable the use of the rule is, i. e. an estimation of the quality of the logical implication asserted by the rule;
- ⟨*if-part*⟩: the Lisp-coded LHS (a Lisp predicate);
- ⟨*then-part*⟩: the Lisp-coded RHS (a Lisp function).

An example of the internal structure of a rule is reported below:

```
((name D97-06)
 (type domain)
 (weight 0.8)
 (if-part
  (lambda (p c)
         (and (objective 'high-precision)
              (forall C (concepts)
                      '(forall C1 (concepts)
                               '(lessp
                                 (abs
                                  (diff
                                    (PC C)
                                    (PC C1)))
                                 100)))
              (setcurrent
                 (exists C (concepts)
                        '(lessp (GL C) 4))))))
then-part
  (lambda
   (p c)
     (activate 'MOST-SPECIFIC-FIRST' on (current)))))
```

In the following, we utilize a simplified description of the rules in order to improve readability. For example, the above rule can be represented in the simplified form as:

D97-06 (0.8)
        IF   all concepts have similar PC
                  there exists a concept C. X with low GL

```
          objective is high-precision
   THEN   activate MOST-SPECIFIC-FIRST on C. X
```

Both syntax and semantics of the language used for the simplified representation of rules are quite intuitive and will be further clarified without a formal definition.

The EKB has a complex organization. It is first partitioned into three parts called types, each type is then further partitioned into blocks, and blocks are finally grouped together to form classes. Let us analyze each one of these components in turn.

Rules may belong to three different types:

- *domain rules,* which are devoted to represent knowledge about the intermediary's activity, and are used to actually carry out the task assigned to the reasoning module. Their RHS may contain both actions and the directives 'activate' and 'terminate' (see Sect. 5). The rule previously discussed (D97-06) is an example of a domain rule.
- *matching rules,* which contain meta-knowledge devoted to reduce the effort involved in the matching phase during operation of the rule-based system. They allow identification of subsets of domain and conflict rules, which are expected to be relevant in the current context, and will be utilized by the interpreter of the production system in the next steps. These subsets are selected through the directive 'use' contained in the RHS of matching rules. The syntax of 'use' is:

  use ⟨domain block⟩ [⟨conflict block⟩],

  where ⟨domain block⟩ and ⟨conflict block⟩ are the names of subsets of domain and conflict rules in the EKB (note that the second argument of 'use' is optional). The following is an example of a matching rule:

  ```
  M65-05 (1.0)
         IF     PAR is not empty
         THEN   use D12  C18
  ```

- *conflict rules,* which contain meta-knowledge devoted to encode criteria for conflict resolution, i.e. for selecting the next rule to be executed from the set of those whose LHS is satisfied in the current context (the conflict set). Conflict rules may contain in the RHS any of the following directives: 'select', 'weigh', 'choose', 'activate', and 'terminate'. The former three directives are used when the conflict set is not empty. Their general form is:

  select ⟨rule name⟩
  weigh
  choose ⟨pattern 1⟩ ⟨pattern 2⟩ . . . ⟨pattern N⟩

  The first directive specifies unconditional selection of the rule referred to by ⟨rule name⟩. The second one causes selection of the rule with highest weight.

The last one chooses the first rule which matches the left-most specified pattern, among ⟨pattern-1⟩, ⟨pattern-2⟩, . . ., ⟨pattern-N⟩. Conflict rules devoted to managing the empty conflict set situation may contain the directives 'activate' or 'terminate' since the others are meaningless.
Note that conflict rules may also contain in the RHS any other kind of actions in addition to the above mentioned directives.
Examples of conflict rules are:

```
C41-02  (0.8)
        IF      CONFLICT-SET is not empty
                objective is high-precision
        THEN    weigh

C41-03  (1.0)
        IF      CONFLICT-SET is empty
                objective is high-recall
                number of concepts in the PIR is small
        THEN    activate GENERALIZE
```

Types are further partitioned into *blocks:* domain, matching, and conflict blocks, respectively. Blocks represent small chunks of knowledge that refer to some specific aspect of the subject domain. Therefore, blocks are expected to be relevant to a well identified class of situations during the operation of the reasoning module.

Since both domain knowledge and meta-knowledge are needed to effectively carry out the tasks involved in the operation of the reasoning module, blocks of different types are grouped together to form larger aggregates called *classes.* A class contains one matching block and one or more domain and conflict blocks. Moreover, matching and conflict rules of a class cannot refer (through the directives 'use' and 'select', respectively) to blocks and rules outside the class. Note that a block may belong to several classes.

Names are assigned to classes, blocks, and rules according to the following rules:

- classes are named with a mnemonic string referring to its meaning (e.g., APPROACH-SELECTION, MOST-SPECIFIC-FIRST, PARALLEL, etc.);
- blocks are named with a string made up of a letter belonging to the set {D, M, C} which indicates the type of the block, and a number which identifies the block in the set of blocks of the same type e.g., D27, CO2, M12, etc.);
- rules are named by concatenating the name of the block to which the rule belongs with a number which identifies the rule inside the block (e.g., D27-01, D27-15, M12-02, etc.).

The structure of the EKB outlined above comprises two basic features:

- knowledge is organized into separate chunks, the classes, each referring to a specific facet of the subject domain, such as a subfield of specialist competence, a problem solving approach, an exceptional situation, a phase of the reasoning process, etc.;
- meta-knowledge is contained in two parts of the representation:
  - the directives appearing in the RHS of domain rules, which basically embody knowledge about which class to use next and how to use it, thus allowing implementation of data-driven switching between classes during system operation;
  - the matching and conflict rules, which contain knowledge about which blocks of domain rules to use within a given class and how to use them, thus allowing selective focusing on the specific knowledge actually relevant to the current context.

The proposed representation model appears to be especially suitable for those domains that are highly structured in nature, and contain a lot of meta-knowledge intermixed with domain knowledge. In such cases, it allows easy and natural knowledge acquisition and representation, since appropriate constructs are available to model a large variety of knowledge structures without the need of an unnatural fragmentation. Moreover, the structure of the EKB allows explicit and effective representation of control knowledge that, otherwise, would be lost or dispersed in a flat, hardly usable representation.

4.3 Problem Internal Representation

The *problem internal representation (PIR)* represents the working memory of the rule-based system. Its content is updated at each step of system operation as a consequence of rule execution. Basically, the PIR is splitted into two parts: the *problem information (PI)* and the *control information (CI).*

The PI part of the PIR contains a structured representation of the available knowledge about the search problem.

The sources of such knowledge are:

- the user (through the understanding and dialogue module), who submits to the system his initial request at the beginning of the session (presearch interview), and answers queries formulated by the system whenever he is engaged in a dialogue;
- the DKB (through the reasoning module) which can supply new information whenever needed during system operation.

The users of the PI are the reasoning module and the formalizer module.

The PI part of the PIR is organized as a frame comprising subframes which, in turn, can contain several slots. The general structure of the PI is the following:

- *Concepts:* a sequence of subframes, each containing the representation of a concept supplied by the user or acquired from the DKB. Each subframe comprises the following slots:
  - *name:* a string built up by "C." followed by a sequential number (e.g., C.1, C.2, C.3, etc.), which identifies the concept in the PIR;
  - *term:* the term (or expression) with which the concept is referred to in the DKB;
  - *PC:* the posting count of the term;
  - *GL:* the generality level of the term;
  - *CT:* the flag (Y, N) denoting whether the term is controlled;
  - *DI:* the interest degree the user has assigned to the term;
  - *related:* the list of names of synonyms, spelling variations, and related keywords entered by the user;
  - *derived:* the list of names of concepts acquired from the DKB during the analysis of the concept.
- *Search-logic:* the logic underlying the current search formulation. It is given through a boolean combination (using the operators OR, AND, NOT) of the names of the concepts actually involved in the search formulation.
- *Objective:* the goal of the search stated by the user. The objective may be high-precision, high-recall, or sample, that denote the desire of obtaining only relevant documents, all relevant documents, and just a small set of relevant documents, respectively[1].
- *Limitations:* a subframe representing the constraints supplied by the user to limit the search. Its slots are:
  - *Language:* the language in which a document must be written in order to be relevant to the search (e.g., English, French, etc.);
  - *date:* a time period the publication date of the document must belong to (e.g., >1975, > =1960 and < =1970, etc.);
  - *Treatment:* the nature of documents to be retrieved (e.g., practical, theoretical, general, formal, etc.).
- *Output:* a subframe containing the specifications of the way in which the retrieved documents are to be supplied to the user. It comprises the following slots:

[1] The definitions of *precision* and *recall* are:

$$\text{precision} = \frac{\text{No. of relevant documents retrieved}}{\text{No. of documents retrieved}}$$

$$\text{recall} = \frac{\text{No. of relevant documents retrieved}}{\text{No. of relevant documents stored in the data base}}$$

- *Format:* e.g., bibliographic reference, abstract, full format, etc.;
- *Max:* the maximum number of items desired;
- *List:* e.g., sorted by year, sorted by author name, not sorted, etc.;
- *Mode:* e.g., online display, offline print, etc.

• *Search-mode:* a flag to be utilized during search strategy design, indicating whether the search has to be carried out on any field of each record in the data base (free-text), or only on those fields containing controlled terminology (controlled vocabulary).

The CI part of the PIR contains information the system uses for its own work, including partial result and internal state variables. Its entries are called *registers* and are created and deleted dynamically by the system.

A general illustration of the structure of CI is not given here, as it mostly concerns system implementation.

5 Reasoning Mechanism

5.1 The Concept of Task

The mode of operation of the reasoning module basically conforms to the general organization of a rule-based system and comprises a recognize-act cycle including the three activities of matching, conflict resolution, and execution [HAYE83]. However, it embodies a novel concept, the *task,* which allows effective use of meta-knowledge, fully exploiting the structured organization of the EKB and of the PIR illustrated in the previous section.

A *task* can be defined as a triple:

T = (CL, PAR, PRED),

where:

- CL is the name of a class of the EKB;
- PAR is a list of concept names belonging to the PIR, called *parameters;*
- PRED is a predicate defined over the PIR, called *termination predicate.*

Both PAR and PRED may be empty.

Tasks are dynamic entities, created at run time as a consequence of a specific directive, 'activate'. Such a directive takes the following form:

activate CL [on PAR] [until PRED],

where the last two components are optional.

The semantics of the directive 'activate' can be stated as follows: use the rules in CL, focusing on the concepts specified in PAR, until PRED evaluates to true. If parameters are omitted the task will operate on all the concepts appearing in the PIR, and when no termination predicate is specified the task will terminate only if the directive 'terminate' is executed.

We denote by *activation* the process of generating a task, putting together a class with a list of parameters and a termination predicate. Similarly, *termination* denotes the fact that a task ceases to exist when its termination predicate becomes true or the directive 'terminate' is executed, and *execution* refers to the sequence of operations necessary to actually run a task.

Henceforth, we will refer to a task through the name of its CL component. PAR and PRED will be explicitly specified whenever necessary to avoid ambiguity.

5.2 Task Execution

We now illustrate how tasks affect the usual recognize-act cycle of the interpreter of a production system. Task execution is the basic mechanism of operation of the reasoning module, whose kernel is the interpreter of the directive 'activate', called the *task interpreter.*

The basic schema of operation of the task interpreter is illustrated in Fig. 2. Let us analyze this schema in detail. The execution of the directive

activate CL on PAR until PRED

causes both the generation of the task T=(CL, PAR, PRED) and a call to the task interpreter for the execution of T. When T is supplied as input to the interpreter it becomes active and its execution starts.

Let us assume that CL has the following structure:

CL=	MB	matching block
	DB ={DB1, DB2,...}	set of domain blocks
	CB ={CB1, CB2,...}	set of conflict blocks

The interpreter will consider in its activity only the concepts contained in PAR, if any; otherwise, the interpreter will operate on the entire PIR. We denote by PIR* the part of the PIR that the interpreter will actually consider during operation.

The interpreter takes first into account the matching block MB of CL and starts the first activity, called *block matching.* This consists of matching the LHS of the rules of MB against PIR*, in order to determine the domain and conflict blocks to be used in the current context. All domain and conflict blocks (belonging to DB and CB, respectively) mentioned in the directive 'use' of the matching

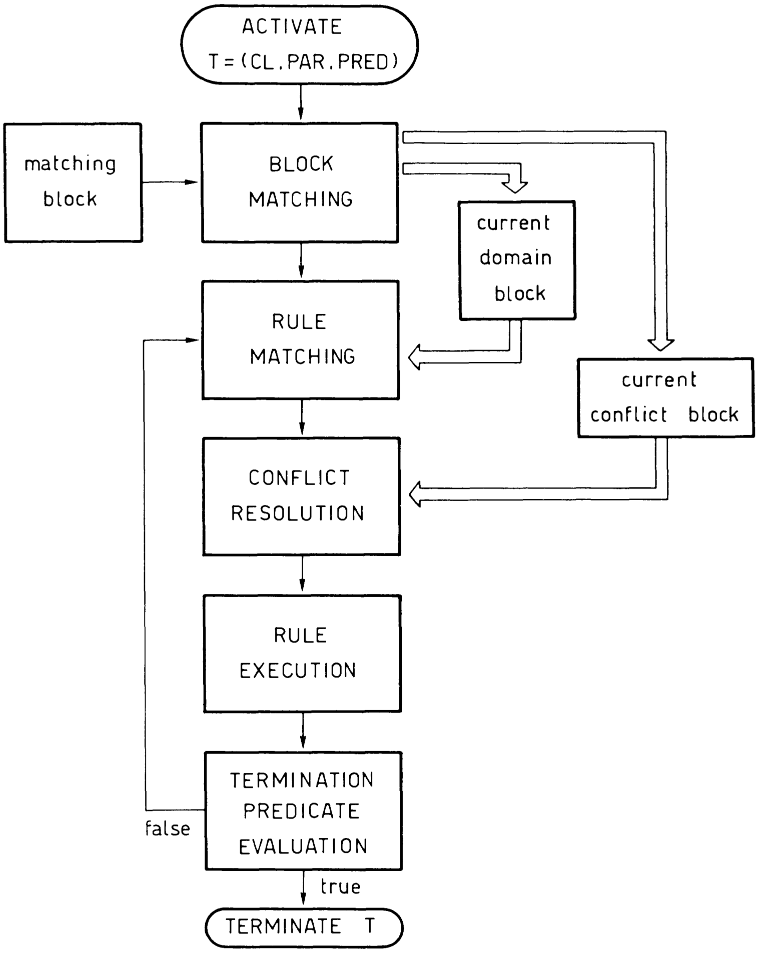

Fig. 2. Schema of operation of the task interpreter

rules whose LHS is satisfied, are merged together to form the current domain and conflict blocks (let us call these DB* and CB*, respectively).

After DB* and CB* have been determined, a usual recognize-act cycle is entered. It comprises three activities:

- *rule matching:* the LHS of the rules in DB* is matched against the current PIR*, and the set of domain rules whose LHS is satisfied is determined (the conflict set).
- *conflict resolution:* the LHS of the rules in CB* is matched against the current PIR*, and the domain rule to be executed next is selected according to the choice determined by execution of the directives appearing in RHS of the con-

flict rules whose LHS is satisfied. If the conflict set is empty, no domain rule can be selected and execution of the directives appearing in the RHS of the conflict rules whose LHS is satisfied will directly determine how task execution will be carried on.

- *rule execution:* the actions and directives appearing in the RHS of the selected domain rule are actually executed and the current PIR* is updated.

If a termination predicate PRED is specified in T, it is now evaluated in relation to the current PIR* *(termination predicate evaluation).* In case it evaluates to true, task execution is terminated; otherwise, control flow is returned to the rule matching activity, and a new recognize-act cycle is started.

Independently of PRED, task execution is also immediately terminated if, during conflict resolution or rule execution, the directive 'terminate' is invoked.

Note that the RHS of both conflict and domain rules can contain the directive 'activate' (possibly in addition to other actions and directives), whose execution causes generation of a new task and a call to the task interpreter. Therefore, if during conflict resolution or rule execution, the directive 'activate' is called upon, the execution of the current task is suspended and a new task becomes active. The suspended task will be resumed, and its execution will continue from the point where it was suspended, after the new task will terminate.

This policy of suspending and resuming task execution is implemented through a stack. The active task is on top of the stack. When a new task is activated, it is pushed on the stack and the point where the current task has been suspended is marked. When a task is terminated it is popped from the stack and the task on top of the stack becomes active again and execution continues from the marked point. If the stack is empty, operation of the task interpreter terminates.

The activity of the task interpreter is initialized by forcing execution of the directive

```
activate MAIN until (user-is-satisfied)
```

that pushes the task MAIN on the stack. The class MAIN contains just one matching rule:

```
M01-01 (1.0)
        IF      true
        THEN    use D01
```

and one domain rule:

```
D01-01 (1.0)
        IF      true
        THEN    activate PRESEARCH-INTERVIEW
                activate APPROACH-SELECTION
                activate RESULT-EVALUATION.
```

The operation of the reasoning module is fully determined by the execution of the task MAIN which controls the correct sequencing of phases in a search session and activates appropriate tasks devoted to actually carrying out the activities proper of each phase.

As a closing remark, we note that the intuitive meaning of task activation is that of switching from one domain of competence to another, focusing on a different facet of the problem and using for this a specialized chunk of knowledge. Tasks can therefore be viewed as specialists (or specific separate expert systems with their own domain knowledge and meta-knowledge), that can be invoked in a data-driven way whenever their competence seems to be useful to further the reasoning process. Each specialist can be activated in several different situations, and can adapt, through its own meta-knowledge (matching and conflict rules), to the specific features of the current context.

The task mechanism shares several features with the blackboard architecture of HEARSAY-III [ERMA81] but activation of specialists is, in our approach, only indirectly data-driven. In fact, for a task to be activated it is necessary that a specific situation is recognized by some rule belonging to another task, and, therefore, specialists are not called upon by data on the blackboard, but they are activated by another specialist (although still in a data-driven way as a consequence of rule execution). Therefore, tasks are not fully independent knowledge sources, but each of them must be knowledgeable of other tasks that it can activate during operation, whenever appropriate. This kind of relationships among tasks greatly reduces the non-determinism of the system operation at the meta-level, thus improving efficiency. This also eliminates the need for having structures such as the strategy knowledge sources of the blackboard model for meta-level planning.

The way tasks are activated can also be compared with the use of meta-rules [DAVI80]. In fact, the directive 'activate' can be viewed as a way of selecting a subset of the rule base on which to focus next. However, the activation of a task is not merely a heuristic-guided selection of the rule to be utilized next, but it implies a change of viewpoint and the decision to use a specific chunk of structured knowledge (both at domain and meta-level) to further the reasoning process.

6 Sample Sessions With IR-NLI

6.1 Structure of a Search Session

In this section two sample search sessions with IR-NLI are presented. Focus is centered on the conceptual aspects of IR-NLI operations, disregarding several marginal details. The examples chosen concern two approaches which are very common in practice: namely, Building Block and Citation Pearl Growing.

Before discussing these two examples in detail, we briefly illustrate the general structure of a search session with IR-NLI, which will serve as a background for the following discussion.

Presearch Interview is the initial phase of a search session, where the user formulates the first version of his request. The goal of this activity is to support the user in explicating his information needs, and to enable the intermediary to clearly understand the precise content and objectives of the user's request.

In IR-NLI the user is asked by the system to state a query in English, specifying the topic it relates to, some keywords and synonyms, the facets which are to be considered, the limitations to be imposed on the search, the desired form of the output results, etc. This activity is entirely carried out by the PRESEARCH-INTERVIEW task, whose goal is that of managing the dialogue with the user during the presearch interview, resorting to the understanding and dialogue module for the linguistics aspects of the communication between the user and the system.

After the user's needs have been definitely acquired, the intermediary has to devise the search strategy. The very first step towards this goal consists in completing the initial formulation of the user's request with additional specific information, that can be obtained from available searching referral aids and can be useful for evaluating the content of the request. The intermediary tackles then the most critical phase of his job, namely *approach identification,* the choice of the most appropriate approach to adopt for an effective management of the design of the search strategy. This choice is done without interacting with the information retrieval system, taking into account the following two elements: the objectives of the user, and the nature and richness of the information supplied by the user or obtainable from available searching referral aids.

In IR-NLI this job is performed by the APPROACH-SELECTION task which operates in two steps. First it extracts from the DKB the posting count, generality level, and controlled term flag of the concept involved in the user's request; then, it selects the approach which seems the most suitable one, according to the information currently available.

Concept Analysis, the central activity of a search session, is then started. This phase has the main goal of expanding (through extraction of new concepts from the DKB) the current formulation of the user's request in order to gather all information needed to construct an effective search strategy.

During this phase, the intermediary designs (either interactively with the information retrieval system or off-line) the structure and content of the search strategy, using appropriate tactics according to the chosen approach. This activity is highly data-driven, as the decisions to be taken and the individual steps to be performed strictly depend on the type and content of the pieces of information already gathered in the previous stages of the analysis.

In IR-NLI this phase is carried out by the specific task implementing the chosen approach, which is directly activated by the APPROACH-SELECTION task and can activate, in turn, all other tasks needed for execution of the relevant tactics.

A crucial problem in this phase is deciding when to stop the analysis: either the data gathered so far are enough for strategy construction or the analysis has evolved to an unsatisfactory point. In both cases we are faced with a judgemental step, which can require interaction with the user.

In the case where concept analysis has succeeded in producing the information needed for strategy construction, the formalizer module is activated, and the final search strategy in the appropriate query language is generated. Otherwise, if concept analysis terminates with a failure, backtracking to a previous step of the analysis, or to the approach identification phase takes place and a different solution is tried.

The last phase of a search session is *result evaluation*. This activity begins after the search has been done and results have been made available to the user. Its purpose is to allow the user to evaluate whether his information needs are fully satisfied and the session can be closed, or the search has to be continued, refined, or even repeated from the beginning with a new request. Usually, the responsibility of result evaluation is entirely left to the user, and the intermediary only supports him in expressing his judgements and formulating appropriate requests to further the search.

In IR-NLI this phase is managed by the RESULT-EVALUATION task, which presents to the user the obtained results and, interactively, supports him in evaluating their relevance.

6.2 An Example With the Building Block Approach

In this section we present a first example of a search session carried out by IR-NLI.

As already pointed out in Sect. 5, the task MAIN, which is the first task to be activated at the beginning of a search session, activates the task PRESEARCH-INTERVIEW to support the user in formulating his request. The first operation that the directive 'activate PRESEARCH-INTERVIEW' performs is block matching. The matching block of the class associated to the task PRESEARCH-INTERVIEW contains, among others, the following rule M02-05, which matches the current PIR and is therefore executed.

```
M02-05 (1.0)
       IF      PIR is empty
               PAR is empty
       THEN    use D01
```

The block D01, addressed by the directive 'use', contains rules that perform the presearch interview.

At this point, the user is invited by the system to submit his request. The dialogue between IR-NLI and the user begins as follows:

```
IR-NLI > Please, state your request!

user   > I would like to have references about computer applications in
         warehousing.
         I am interested only in English documents not older than three
         years.
```

The first step of the presearch interview is completed; the understanding and dialogue module analyzes the request and extracts the information relevant to the search. The first PIR (PI part) obtained is:

```
concepts:     name = C.1
              term = computer applications

              name = C.2
              term = warehousing

limitations:  language = English
              date > = 1981
```

The user has now to supply further information to better specify his needs: in the block D01 there are rules which can manage a bounded scope dialogue for collecting new useful information from the user and expanding the PIR. Some examples of dialogue rules are:

```
D01-07 (1.0)
       IF     concepts is empty
       THEN   get concepts

D01-16 (0.7)
       IF     PIR has been filled
       THEN   terminate
```

where 'get' is a Lisp function which acquires data from the user through a fixed dialogue.

The effect of these rules is the following interaction with the user:

```
IR-NLI > Please, specify the scope of the search:
         - comprehensive: retrieval of all relevant items (high-recall)
         - limited: retrieval of only relevant items (high-precision)
         - sample: retrieval of some relevant item
user   > comprehensive

IR-NLI > Now, please, specify the treatment desired (general, practical,
         theoretical, . . .)
```

```
user    > practical

IR-NLI  > Output desired:
          - offline print, online display, . . .
          - format: bibliographic reference, abstract, full format, . . .
          - sorted, . . .
          - maximum number of items desired

user    > full format, offline print, not sorted, with at most 50 postings

IR-NLI  > Please, specify synonyms, spelling variations, related key-
          words of each concept contained in your request:

          C.1 computer applications:

user    > management information systems

IR-NLI  > C.2 warehousing

user    > none

IR-NLI  > Please, supply any particular point of view or specific aspect to
          take into account (design, performance, reliability, security,
          . . .)

user    > cost/benefit evaluation

IR-NLI  > For each concept please specify the degree of relevance to the
          search (score from 1 to 10)

IR-NLI  > C.1 computer applications

user    > 10

IR-NLI  > C.2 warehousing

user    > 10

IR-NLI  > C.3 cost/benefit evaluation

user    > 5
```

Rule D01-07 allows identification of concepts and related terminology and, moreover, it constructs the search logic. The search-logic reflects the internal logical structure of the initial request.

Once all information that has to be collected in the presearch interview has been supplied by the user (and inserted in the PIR), rule D01-16, above presented, is executed. Therefore, the task PRESEARCH-INTERVIEW terminates, and control is returned to the task MAIN. The updated PIR is now the following:

```
concepts:    name = C.1
             term = computer applications
```

	DI = 10
	related = C. 3

	name = C. 2
	term = warehousing
	DI = 10

	name = C. 3
	term = management information systems

	name = C. 4
	term = cost/benefit evaluation
	DI = 5

search-logic:	(AND (OR C. 1 C. 3) C. 2 C. 4)
objective:	high-recall

limitations:	language = English
	date > = 1981
	treatment = practical

output:	format = full format
	max = 50
	list = not sorted
	mode = offline print

APPROACH-SELECTION is the next task activated by MAIN. The first step carried out is to get additional information from the DKB. This is accomplished by the task FILL-IN, which is activated if there is at least a concept whose slots 'PC', 'GL', and 'CT' are empty. The actions that FILL-IN takes are straightforward: for each concept considered, it transfers the values of the slots from the DKB to the PIR.

The portion of DKB accessed by FILL-IN in our example is illustrated in Fig. 3 and 4. In these figures, the DKB is represented as a labeled network where two types of nodes appear. The 'closed' ones represent the nodes actually examined by the system in the example, while the 'open' nodes represent concepts that are not taken into account.

After the execution of FILL-IN, the PIR concerning the concept C. 1 contains:

concepts:	name = C. 1
	term = computer applications
	PC = 12004
	GL = 8
	CT = Y
	DI = 10
	related = C. 3

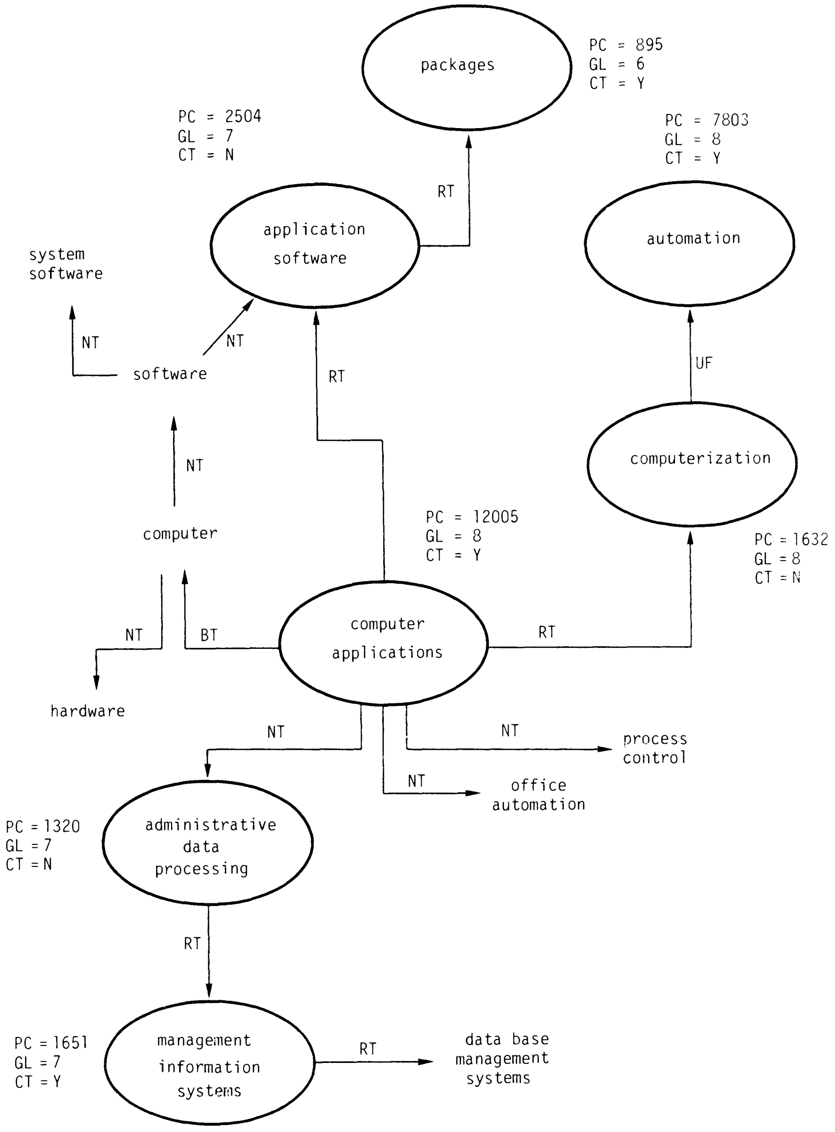

Fig. 3. DKB fragment around the term COMPUTER APPLICATIONS

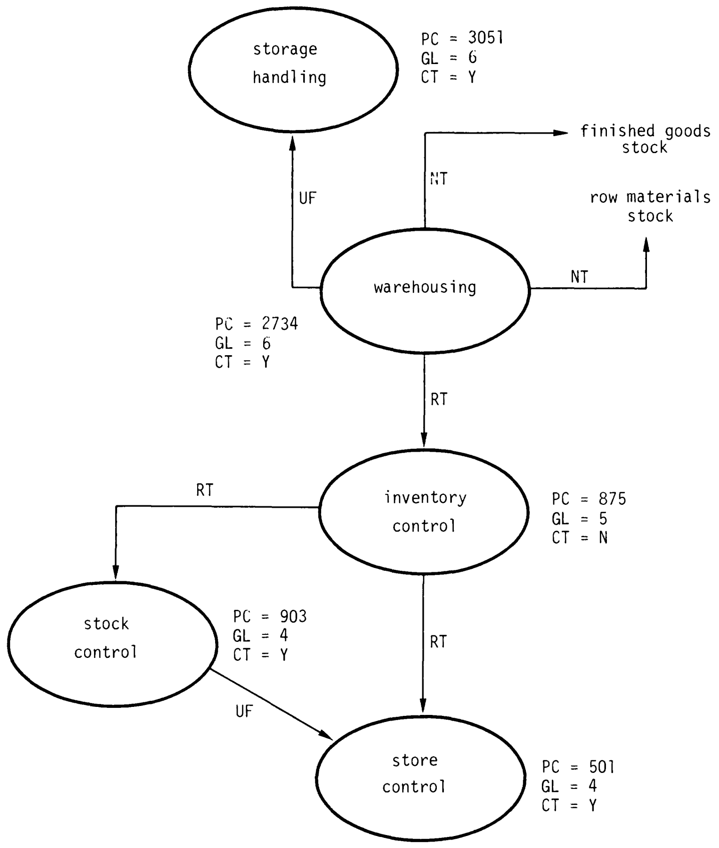

Fig. 4. DKB fragment around the term WAREHOUSING

Similar updates are performed on the other concepts. The second step of APPROACH-SELECTION is the choice of the approach. The domain rule executed is the following one:

```
D12-02 (0.9)
    IF      number of concepts is low
            all concepts have high PC
            all concepts have similar GL
    THEN    activate BUILDING-BLOCK
            activate SEARCH
            terminate
```

The number of concepts is considered to be low when it does not exceed a given constant. Moreover, a concept has high PC when its posting count is greater than a given threshold. Finally, all concepts have similar GL when for any pair of concepts the difference of their generality levels does not exceed a fixed constant.

In the RHS of rule D12-02, there is a sequence of three directives; the first two ones are devoted to activate tasks, the last one to stop the APPROACH-SELECTION task. BUILDING-BLOCK is devoted to implement the BBL approach and SEARCH both performs the synthesis of the search strategy and submits it to the information retrieval system.

Although not shown here, BUILDING-BLOCK first executes a matching rule which selects the domain block D07 and the conflict block C01, and second it executes the rules contained in block D07.

The situation we are considering in our example yields to a conflict resolution activity handled by conflict rules.

In fact, after the rule matching phase, the current conflict set contains the following three rules:

```
D07-01 (0.6)
       IF     objective is high-recall
              search-mode is empty
       THEN   set search-mode to free-text

D07-11 (0.8)
       IF     objective is high-recall
              there is a concept C not yet compiled
              number of terms in C is low
       THEN   activate COMPILE on C

D07-13 (0.8)
       IF     objective is high-recall
              there is a concept C not yet expanded
              number of terms in C is low
       THEN   activate PARALLEL on C
                     until number of terms in C is high
```

where COMPILE refers to the process of specifying spelling variations or truncations of a term, according to the tactics 'rearrange', 'respace', 'respell', and 'fix', while PARALLEL refers to the process by which the terminology related to a term is increased through the tactics 'parallel', 'relate', 'sibling', etc. The conflict rule which matches this conflict situation is:

```
C01-02 (1.0)
       IF     CONFLICT-SET is not empty
              objective is high-recall
       THEN   choose PARALLEL
                     COMPILE
                     GENERALIZE
                     REDUCE
```

whose meaning is trying to solve the conflict by choosing the rule which contains the first-mentioned pattern. In fact, in this situation, due to the high-recall objective, it is better to consider the tactic 'parallel' rather than others, such as 'generalize' or 'reduce'.

In our case rule D07-13 is executed which activates the task PARALLEL, devoted to the expansion of the terminology of the concepts specified in PAR. The expansion is carried out through the application of tactics such as: 'relate', 'sibling', 'generalize', 'parallel', etc.

The following matching rule is first executed:

```
M22-02 (1.0)
        IF      PAR is not empty
        THEN    use D26  C03
```

The rules of the block D26 whose LHS is satisfied in the current PIR are:

```
D26-01 (0.9)
        IF      objective is high-recall
                exists C in PAR with low GL
        THEN    activate RELATE on C
D26-02 (0.7)
        IF      objective is high-recall
                all concepts in PAR have low number of terms
        THEN    activate PARALLEL on terms derived from PAR
```

The conflict is resolved by the rule:

```
C03-04 (1.0)
        IF      CONFLICT-SET is not empty
                objective is high-recall
        THEN    choose RELATE
                       PARALLEL
                       GENERALIZE
                       COMPILE
                       SIBLING
```

which results in the selection of rule D26-01. In this case C is instantiated by C. 2 and the task RELATE is activated, which accesses the DKB and expands the terminology of C. 2 by including terms linked to C. 2 through BT arcs.

After the execution of the task RELATE the portion of PIR concerning C.2 is:

```
concepts:     .
              .
              .
              name = C.2
              term = warehousing
              PC = 2730
              GL = 6
              CT = Y
              DI = 10
              derived = C.5
              .
              .
              .
              name = C.5
              term = inventory control
              PC = 3051
              GL = 7
              CT = Y
              .
              .
              .
```

When RELATE terminates, PARALLEL resumes control. Since its termination predicate (number of terms in PAR is high) has not yet been verified, it continues to match its rules against the PIR and to execute them. In a similar way, other terms are acquired by PARALLEL for the concept C.2.

When the task PARALLEL terminates, control is again transferred to BUILDING-BLOCK which again will match domain rules of block D07 containing the activation of PARALLEL for other concepts. Other tasks are later activated by BUILDING-BLOCK; among these we mention the task COMPILE, charged with the compilation of terms. Some of its domain rules are:

```
D29-02 (1.0)
       IF     HYPHEN is in PAR
       THEN   substitute HYPHEN with SPACE

D29-03 (1.0)
       IF     HYPHEN is in PAR
       THEN   substitute HYPHEN with NULL

D29-07 (0.9)
       IF     '-ing' is in PAR
       THEN   truncate '-ing'
```

```
D29-08 (0.9)
       IF     '-ation' is in PAR
       THEN   truncate '-ation'

D29-12 (0.8)
       IF     PAR is plural
       THEN   truncate PAR to singular
```

Operation continues with execution of all the possible domain rules, and when BUILDING-BLOCK finishes its activity, it terminates and the resulting PIR is the following:

```
concepts:    name = C.1
             term = computer.applic*
             PC = 12004
             GL = 8
             CT = true
             DI = 10
             related = C.3
             derived = C.6 C.7 C.8 C.9 C.10

             name = C.2
             term = warehous*
             PC = 2730
             GL = 6
             CT = true
             DI = 10
             derived = C.5 C.11 C.12 C.13

             name = C.3
             term = management.information.system*

             name = C.4
             term = cost*.benefit*.evalu*

             name = C.5
             term = inventory.control

             name = C.6
             term = applic*.software

             name = C.7
             term = computeriz*

             name = C.8
             term = administrative.data.process*

             name = C.9
             term = autom*

             name = C.10
             term = package*
```

```
               name = C.11
               term = storage.handl*
               name = C.12
               term = stock*.control
               name = C.13
               term = stor*.control
search-logic:  (AND (OR C.1 C.3 C.6 C.7 C.8 C.9 C.10)
                    (OR C.2 C.5 C.11 C.12 C.13)
                    C.4)
objective:     high-recall
limitations:   language = English
               date > = 1981
               treatment = practical
output:        format = full format
               max = 50
               list = not sorted
               mode = offline print
search-mode:   free-text
```

After the termination of BUILDING-BLOCK, control is resumed by rule D12-02 of the task APPROACH-SELECTION. Such a rule activates then the task SEARCH which invokes the *formalizer module* to construct the search strategy from the final version of the PIR. In our example, the following program is generated (in a dialect of EUROLANGUAGE):

```
 1. FIND COMPUTER.APPLIC*
 2. FIND MANAGEMENT.INFORMATION.SYSTEM*
 3. FIND APPLIC*.SOFTWARE
 4. FIND COMPUTERIZ*
 5. FIND ADMINISTRATIVE.DATA.PROCESS*
 6. FIND AUTOM*
 7. FIND PACKAGE*
 8. S=1 OR S=2 OR S=3 OR S=4 OR S=5 OR S=6 OR S=7
 9. FIND WAREHOUS*
10. FIND INVENTORY.CONTROL
11. FIND STORAGE.HANDL*
12. FIND STOCK*.CONTROL
13. FIND STOR*.CONTROL
14. FIND S=9 OR S=10 OR S=11 OR S=12 OR S=13
15. FIND COST*.BENEFIT*.EVALU*
16. FIND S=8 AND S=14 AND S=15
17. LIMIT S=16/LANG=ENGLISH
18. LIMIT S=18/YEAR>=1981
19. LIMIT S=17/TREAT=PRACTICAL
20. PRINT F1, 1 . . . 50
```

Afterwards, SEARCH submits this program to the information retrieval system which will return the actual results of the search. SEARCH terminates and causes termination of APPROACH-SELECTION. Control is given back to MAIN which activates RESULT-EVALUATION. This task outputs first a ranked list of the retrieved items, and, second, it interacts with the user who supplies the relevance evaluation of the documents. If the results satisfy user's needs, the search session concludes; otherwise, the process continues by repeating the operations specified in the task MAIN.

6.3 An Example With the Citation Pearl Growing Approach

The second example we present deals with the Citation Pearl Growing approach.

The first steps of IR-NLI are just the same as in the previous example: first task MAIN is activated and then task PRESEARCH-INTERVIEW performs the presearch interview.

The initial query is the following one:

IR-NLI > `Please, state your request!`

user > `I would like to have references about cooperative man-machine interfaces for online document retrieval. I am interested in English papers.`

The presearch interview continues then as follows:

IR-NLI > `Please, specify the scope of the search:`
`- comprehensive: retrieval of all relevant items (high-recall)`
`- limited: retrieval of only relevant items (high-precision)`
`- sample: retrieval of some relevant item`

user > `limited`

IR-NLI > `Now, please specify the treatment desired (general, practical, theoretical, ...)`

user > `general`

IR-NLI > `Output desired:`
`- offline print, online display, ...`
`- format: bibliographic reference, abstract, full format, ...`
`- sorted, ...`
`- maximum number of items desired`

user > `full format, online display, sorted by year in ascending order`

IR-NLI > `Please, specify synonyms, spelling variations, related keywords of each concept contained in your request:`

```
IR-NLI  > C.1 document retrieval
user    > none
IR-NLI  > C.2 man-machine systems
user    > none
IR-NLI  > Please, supply any particular point of view or specific aspect to
          take into account (design, performance, reliability, security,
          ...)
user    > design criteria
IR-NLI  > For each concept please specify the degree of relevance to the
          search (score from 1 to 10)
IR-NLI  > C.1 document retrieval
user    > 10
IR-NLI  > C.2 man-machine systems
user    > 10
IR-NLI  > C.3 design criteria
user    > 3
```

PRESEARCH-INTERVIEW terminates as usual when all the possible slots of the PIR are filled with data provided by the user. Control is given back to MAIN which activates task APPROACH-SELECTION.

Task FILL-IN is then activated in order to access the DKB. It produces the following PIR:

```
concepts:    name = C.1
             term = document retrieval
             PC = 3175
             GL = 7
             CT = N
             DI = 10

             name = C.2
             term = man-machine systems
             PC = 512
             GL = 8
             CT = N
             DI = 10

             name = C.3
             term = design criteria
             DI = 3
```

search-logic:	(AND C.1 C.2 C.3)

limitations:	language = English
	treatment = practical

output:	format = full format
	list = sorted by year in ascending order
	mode = online display

objective:	high-precision

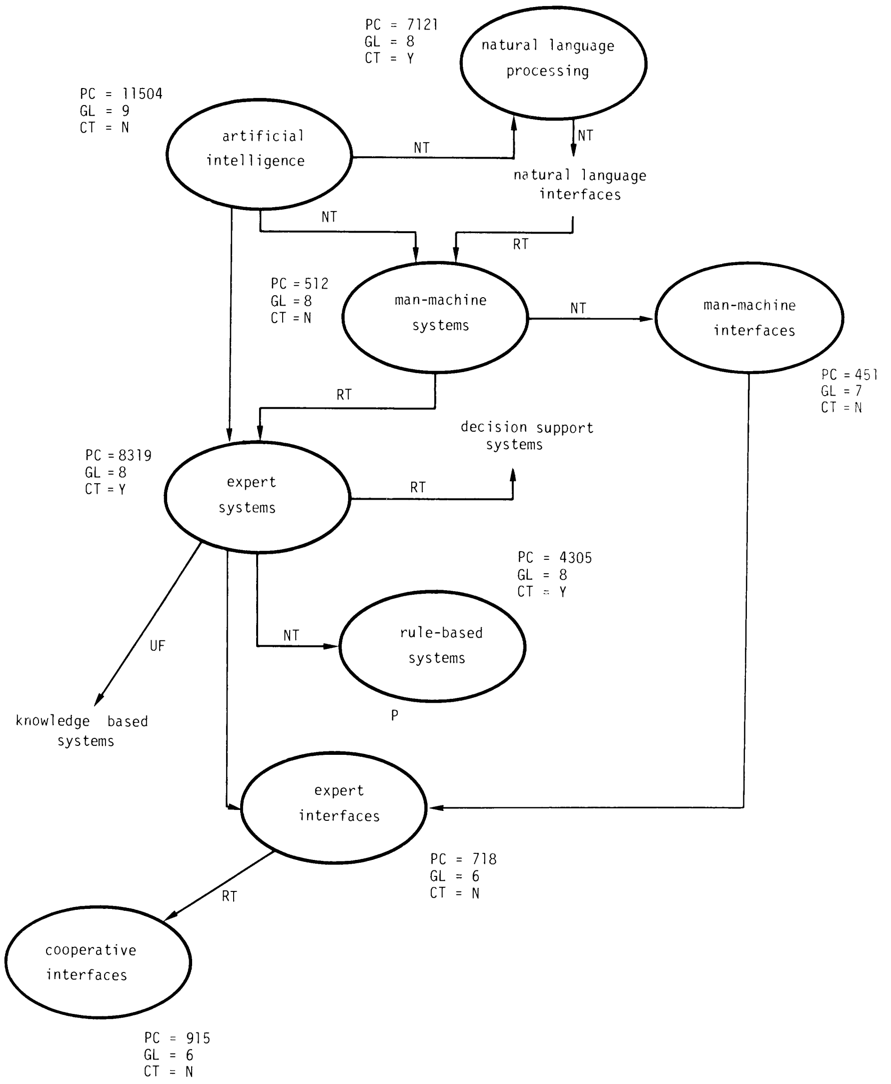

Fig. 5. DKB fragment around the term MAN-MACHINE SYSTEMS

Successively other domain rules, devoted to the choice of the approach, are matched against the current PIR. In particular, the following rule is executed:

```
D12-06 (0.9)
       IF     all concepts have loose connection with DKB
       THEN   activate CITATION-PEARL-GROWING.
```

By the expression loose connection we mean that the number of arcs in the DKB leaving the node representing a concept is low (bounded by a constant). This corresponds to the situation arising when the intermediary is poorly acquainted with the subject proposed by the user, and decides to follow the CPG approach.

The parts of DKB relevant in this context are shown in Fig. 5 and 6.

With the execution of rule D12-06, the task CITATION-PEARL-GROWING is activated. The goal of this task is to enlarge, when possible, the current PIR by trying tasks such as PARALLEL, GENERALIZE, SPECIFY, etc., and afterwards performing a first tentative search.

The results are collected and presented to the user who will choose interesting citations. Task CITATION-PEARL-GROWING will cease to exist when the number of concepts so discovered exceeds a threshold value.

However, since the CPG approach has been selected because of a 'poor' DKB, the results of CITATION-PEARL-GROWING are not expected to be very effective, leading only to a limited increase of terminology.

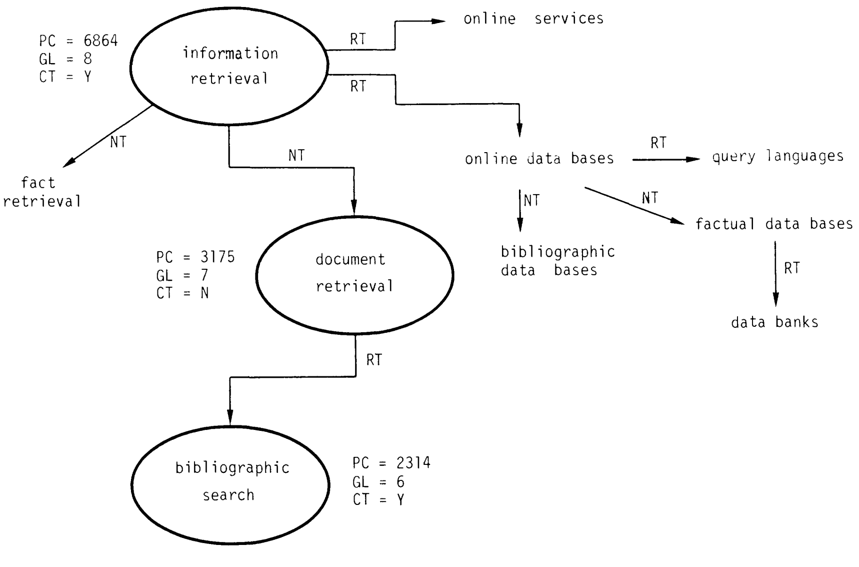

Fig. 6. DKB fragment around the term DOCUMENT RETRIEVAL

In fact, only two terms are found (bibliographic search, expert systems) and are later inserted in the PIR as is shown below:

```
concepts:      name = C.1
               term = document.retrieval
               PC = 3175
               GL = 7
               CT = N
               DI = 10
               derived = C.4

               name = C.2
               term = man-machine.system*
               PC = 512
               GL = 8
               CT = N
               DI = 10
               derived = C.5

               name = C.3
               term = design.criteria
               DI = 3

               name = C.4
               term = bibliographic.search
               PC = 2314
               GL = 6
               CT = Y

               name = C.5
               term = expert.system*
               PC = 8314
               Gl = 7
               CT = Y

search-logic:  (AND (OR C.1 C.4) (OR C.2 C.5) C.3)

limitations:   language = English
               treatment = practical

output:        format = full format
               list = sorted by year in ascending order
               mode = online display

objective:     high-precision

search-mode:   free-text
```

CITATION-PEARL-GROWING now activates the task SEARCH, resulting in the following search strategy:

```
 1. FIND DOCUMENT.RETRIEVAL
 2. FIND BIBLIOGRAPHIC.SEARCH
 3. FIND S=1 OR S=2
 4. FIND MAN-MACHINE.SYSTEM*
 5. FIND EXPERT.SYSTEM*
 6. FIND S=4 OR S=5
 7. FIND DESIGN.CRITERIA
 8. FIND S=3 AND S=6 AND S=7
 9. LIMIT S=8/LANG=ENGLISH
10. LIMIT S=9/TREAT=PRACTICAL
11. PRINT F1
```

Such a search strategy is entered to the information retrieval system and the results are collected. This causes the termination of SEARCH, which in turn leads to the termination of both CITATION-PEARL-GROWING and APPROACH-SELECTION. MAIN activates then task RESULT-EVALUATION that will help the user in the evaluation of the documents extracted from the data base. From these documents the user can identify some new pertinent terms and meaningful citations. The following piece of dialogue supports the user in this job:

```
IR-NLI > Are you satisfied of the documents I just presented?

user   > not completely

IR-NLI > Did you find anything interesting in the above listed docu-
         ment(s)?

user   > yes

IR-NLI > Do you think you have to reformulate your request?

user   > no

IR-NLI > Then, for each concept, give me please some new key-words and/or
         synonyms:

IR-NLI > C.1 document retrieval

user   > information search
         fact retrieval

IR-NLI > C.2 man-machine system

user   > natural language processing
         expert interfaces
         cooperative interfaces
```

FILL-IN is again activated by RESULT-EVALUATION and extracts new information from the DKB, yielding the following PIR:

```
concepts:   name = C.1
            term = document.retrieval
            PC = 3175
            GL = 7
            CT = N
            DI = 10
            related = C.6 C.7
            derived = C.4

            name = C.2
            term = man-machine.system*
            PC = 512
            GL = 8
            CT = N
            DI = 10
            related = C.8 C.9 C.10
            derived = C.5

            name = C.3
            term = design.criteria
            DI = 3

            name = C.4
            term = bibliographic.search
            PC = 2314
            GL = 6
            CT = Y

            name = C.5
            term = expert.system*
            PC = 8319
            GL = 7
            CT = N

            name = C.6
            term = information.search
            PC = 3128
            GL = 8
            CT = N

            name = C.7
            term = fact.retrieval

            name = C.8
            term = natural.language.process*
            PC = 3108
            GL = 8
            CT = Y
```

	name = C.9
	term = expert.interface*
	name = C.10
	term = cooperative.interface*
search-logic:	(AND (OR C.1 C.4 C.6 C.7)
	(OR C.2 C.5 C.8 C.9 C.10)
	C.3)
limitations:	language = English
	treatment = practical
output:	format = full format
	list = sorted by year in ascending order
	mode = online display
objective:	high-precision
search-mode:	free-text

RESULT-EVALUATION returns control to MAIN. The system now evaluates the termination predicate of MAIN, which results to be false, due to the previous dialogue with the user. The system is therefore again in the initial state, firing thus the domain rule D01-01. This will activate first task PRESEARCH-INTERVIEW, that will not lead to any effect since the PIR is already expanded, and, second, APPROACH-SELECTION. This task will find the PIR suitable for the selection of building block approach, and henceforth, operations will continue in a way analogous to the preceding example.

7 Learning as a Tool for Improving Interface Capabilities

As in all of the fields where intellectual activity is involved, a real expert in the area of expert interfaces and, more specifically, intelligent information retrieval should be able to acquire new knowledge without an enormous effort by refining his skill through experience. Both of these capabilities belong to the sphere of learning [CARB83a, CARB83b], but they are featured by quite different issues and mechanisms:

- the *knowledge acquisition* process allows the expert system to directly and autonomously acquire new knowledge from the outside world, generally expressed in a form which is different from that which it uses internally. This requires that the knowledge acquisition process includes a kind of knowledge compilation from an external form to an internal one. Moreover, knowledge acquisition requires the basic capability of estimating the value of available knowledge and its potential usefulness, in order to choose the pieces of knowledge which are worth to be acquired.

- the *skill refinement* capability allows the expert to improve its performance with experience. Usually, in human experts, experience gathering is a natural consequence of practice, and leads to the indirect acquisition of new meta-knowledge that can be used either to improve efficiency in doing routine jobs or to discover effective strategies for addressing new tasks.

None of the above capabilities is present in IR-NLI. Knowledge acquisition does not take place automatically, as the system cannot acquire new knowledge by itself: knowledge can only be inserted by the designer by directly loading it. Moreover, the lack of any skill refinement capability makes IR-NLI performance quite limited: its behavior is fixed by the designer and its performance can not improve with use.

Of the two limitations mentioned above, the latter is far more important. In fact, while the first deals with the acquisition of domain knowledge, the second basically concerns the acquisition of new meta-knowledge, which is much more difficult to elicit and can hardly be represented in a fixed complete way at the moment of system design.

The extension of the capabilities of IR-NLI (more specifically of the reasoning module) to include some skill refinement capability can focus on three basic issues:

1. improving system efficiency in addressing situations already faced in the past;
2. improving system efficiency in addressing situations similar to other ones already solved in the past;
3. improving system capability of facing new critical situations, i.e., that cannot be solved by means of usual reasoning mechanisms.

All of the issues require that the *history* of the system operation is stored in a long-term memory, which can be structured as a sequence of records, each one representing a situation, the solution adopted, and an evaluation of the resulting performance. This knowledge is already available in the current organization of the reasoning module (namely: the PIR, the sequence of task activations, and the evaluation supplied by the user). It only needs to be appropriately represented and stored.

Once the history of the system is available, the implementation of the above learning capabilities will depend on the different ways in which this knowledge can be used.

The first kind of learning only requires *matching* capabilities between the current situation and those stored in history records. If past cases are found to be identical to the present one (i.e., having the same PIR structure: number of concepts, attributes, relations, objectives, etc. - clearly not the same content) and to have a positive performance value, its solution is simply extracted from the history record and directly utilized.

The system does not improve its performance (i.e., ability to face and solve search problems) but can save search effort and, therefore, it can be more efficient. In other words, it provides the same performance it would have produced anyway, without re-inventing a new one, but simply recalling it from memory.

The second kind of learning requires far more skilled capabilities. First of all, the system must be capable of managing *partial matching,* in order to retrieve from the history situations similar to the current one with a positive performance value. Once such a record is found, the system has to apply some *transformation* in order to adapt the solution of the history case to fit the present one. In doing this, it must carefully take into account the difference that partial matching has shown between the present and the past case. If the transformation fails, the search for other similar cases in the history has to be continued.

Similarly to the previous case, the system does not improve performance, but only tries to be more efficient. This issue clearly depends on the assumption that partial matching plus transformation require less effort than the searching needed to face the current situation anew. We note that this kind of learning, although generally much more costly than the first kind, offers a basic advantage: not only can it contribute to the effective solution of the current situation, but it provides a new meaningful record to be included in the history for possible future use. Therefore, it produces new experience.

The third kind of learning refers to a still more ambitious capability. Similar to the previous case, the system retrieves from the history a set of cases similar to the current one, but for none of them a transformation can be successfully applied. The system has to resort, in this case, to a *synthesis* capability. Taking into account the results of partial matching, the system first decomposes the situations and solutions of the similar cases, extracted from the history, into a collection of situation-solution fragments. It then selects a subset of fragments which are expected to be relevant to the solution of the current situation, and tries to assemble the chosen solution fragments into a new solution. This kind of learning is generally very expressive and can be applied only in critical situations where usual reasoning mechanisms fail. The central issue here is not improving efficiency, but providing a new performance, and, therefore, substantially contributing to the extension of system capabilities and experience.

It is worth noting that the history stored in the long-term memory, usually gathered during system operation, can also be provided directly by human experts as a collection of meaningful cases. This provides a very interesting way of instructing the system, without the need for explicitly eliciting, structuring, and representing meta-knowledge.

The three cases of learning discussed above all rely on the direct use of experience, and do not provide any kind of real acquisition of new meta-knowledge. They enable the system to use knowledge on past cases effectively whenever appropriate to face specific situations, but they do not change its general mode of operation. In other words, they can derive from experience new skill, useful in a particular case, but they do not perform any substantial amount of induction to derive from this specific skill rules of general concern.

Therefore, these three cases of learning may be viewed as a first step towards a more ambitious kind of learning. This would enable the system to elicit the general aspects of the specific situations it encounters during operation, and to code them into new knowledge structures, usable directly in future operation without the need of resorting to the history.

Inductive learning [MICH83] can be quite naturally implemented in IR-NLI through a mechanism that can manipulate tasks. The basic operations it must be able to perform on tasks include:

- *Refinement:* from an existing task new more definite (and, generally, less non-deterministic) tasks are generated, which apply to a smaller number of specific situations;
- *Composition:* two or more tasks are merged to form a single larger task;
- *generalization:* conditions for activation of a task are made less restrictive, so that the task applies to a larger number of situations;
- *Aggregation:* rule that are usually applied in sequences within a task are composed into a single larger rule (or task), thus reducing non-determinism of task execution;
- *Creation:* a new task is constructed by the system.

Note that the above operations generally involve not only the task which they apply to, but also all related tasks that refer to it through the directive 'activate', as, generally, new tasks require new activation conditions.

Presently, little experimentation with the first three kinds of learning has been carried out, while implementation of inductive learning has only been planned as a future activity.

8 Conclusion

In this chapter, we have addressed two main issues; the definition of the new concept of *expert interface,* and the design of a specific rule-based reasoning mechanism, centered on the notion of *task,* suitable to model the behavior of an intelligent intermediary.

These concepts have been practically evaluated in the development of the IR-NLI prototype for intelligent information retrieval, that is presently running on a VAX-11/780 (written in Franz LISP). The experimental work carried out with IR-NLI has revealed several directions for future work. In addition to the major issue of learning already discussed in the previous section, we mention here three basic topics.

A first point concerns the study of a possible enrichment of the DKB with *semantic relationships* embedding world knowledge on the subject domain. Such an extension of the DKB, which presently only contains a few basic terminologic relationships between terms, can be the basis for a more skilled concept analysis and a better elicitation of the information needs of the user. The adoption of a simple kind of semantic net (comprising entities, attributes, and relationships) has already been attempted, and seems promising in several critical situations where the user is not a specialist of the subject domain and can express his needs only in a naive and poorly specific way.

A second topic which is partially connected to the extension of the DKB above discussed, deals with the possibility of computing, for each retrieved document, a *relevance factor*. This should express how close to the information needs of the user a document is expected to be. Relevance factors can serve two purposes:

- supporting the user in evaluating the results of a search;
- offering the user the possibility of constraining the search by specifying a range of acceptable relevance factors for the desired documents.

The design of a mechanism for the computation of relevance factors poses several non-trivial problems and requires precise modeling of such concept as information needs and relevance of a document, that directly refer to the foundations of document indexing and retrieval.

A last issue concerns the extension and generalization of the *task mechanism*, which has been designed to meet the specific requirements of an expert interface, but embodies several concepts about representation and use of meta-knowledge that can be the basis for a wider study. A preliminary work in this direction [GUID84] has produced a first outline of a novel expert system architecture supporting a generalization of most of the features of the task mechanism.

References

[ALLE83], [BATE79], [CARB83a], [CARB83b], [CODD74], [COHE82], [DAVI80], [DEFU84], [DOSZ79], [ERMA81], [GREE63], [GUID82], [GUID83a], [GUID83b], [GUID84], [HAYE79], [HAYE83], [HEND78c], [KAPL83], [LANC79], [MARC81a], [MARC81b], [MEAD81], [MICH83], [POLL81], [POLL82], [REIC84], [REIT83], [SALT83], [WATE78], [WILE83], [WOOD72]

References

[AHOU72]
Aho AV and Ullman JD (1972). *The Theory of Parsing, Translation, and Compiling,* Englewood Cliffs, NJ: Prentice-Hall

[AIC82]
Artificial Intelligence Corporation (1982). Intellect Query System Reference Manual, Waltham, Mass.: Artificial Intelligence Corporation

[ALLE83]
Allen J (1983). Recognizing Intentions from Natural Language Utterances, in Brady M and Berwick RC (Eds.), *Computational Models of Discourse,* pp. 107-166, Cambridge, Mass.: MIT Press

[ANSI75]
American National Standards Institute, Study Group on Data Base Management Systems (1977). ANSI/X3/SPARC Interim Report on Data Base Management Systems, FDT (Bulletin of ACM-SIGMOD), 7 (2)

[APPE82]
Appelt DE (1982). Planning Natural Language Utterances to Satisfy Multiple Goals, Report STAN-CS-82, Stanford University

[ASTR76]
Astrahan MM, Blasgen MW, Chamberlin DD, Eswaran KP, Gray JN, Griffiths PP, King WF, Lorie RA, McJones J, Mehl JW, Putzolu GR, Traiger IL, Wade BW and Watson V (1976). System R: Relational Approach to Database Management, *ACM Transactions on Database Systems,* 1 (2), pp. 97-137

[BALL83]
Ballard BW (1983). On the Need for Careful Description of NL Prototypes, *American Journal of Computational Linguistics,* 9 (1), pp. 23-24

[BARB83]
Barber RE and Lucas HC (1983). System Response Time, Operator Productivity, and Job Satisfaction, *Communications of the ACM,* 26 (11), pp. 972-986

[BARR81]
Barr A and Feigenbaum EA (Eds) (1981). *The Handbook of Artificial Intelligence,* London

[BATE78]
Bates MJ. (1978). The Theory and Practice of Augmented Transition Network Grammars, in [BOLC78], pp. 119-159

[BATE79]
Bates MJ (1979). Information Search Tactics, *Journal of the American Society for Information Science,* 30, pp. 205-214

[BATE83]
Bates MJ, Bobrow RJ (1983). A Transportable Natural Language Interface for Information Retrieval, *Proceedings 6th ACM-SIGIR Conference,* Washington, DC

[BECK69]
Becker AL and Arms DG (1969). Prepositions as Predicates, Papers from the Fifth Regional Meeting of the Chicago Linguistic Society, April, Chicago, Illinois University of Chicago

[BERG82]
Bergmann H (1982). Lemmatisierung in HAM-ANS, Memo ANS-10, Universität Hamburg, Forschungsstelle für Informationswissenschaft und Künstliche Intelligenz

[BERT81]
Bertrand O, Daudenarde JJ and du Castel B (1981). User Language Generator. Program Description/Operation Manual, Paris: IBM France

[BOBR84]
Bobrow R (1984). Personal Communication Re: A New Implementation of KL-ONE, May

[BOLC78]
Bolc L (Ed) (1978). *Natural Language Communication with Computers,* Berlin: Springer

[BOLC80]
Bolc L (Ed) (1980). *Natural Language Based Computer Systems,* Munich: Hanser

[BOLC84]
Bolc L (Ed) (1984). *Natural Language Generation Systems,* Berlin: Springer

[BRAC79a]
Brachman RJ (1979). On the Epistemological Status of Semantic Networks, in Findler NV (Ed), *Associative Networks - the Representation and Use of Knowledge in Computers,* pp. 3-50. New York: Academic Press

[BRAC79b]
Brachman RJ, Bobrow RJ, Cohen PR, Klovstad JW, Webber BL and Woods WA (1979). Research in Natural Language Understanding - Annual Report 1 Sept 78-31 Aug 79, Technical Report 4274, Cambridge, Mass.: Bolt Beranek and Newman

[BRAU76]
Braun S and Schwind C (1976). Automatic Semantics-Based Indexing of Natural Language Texts for Information Retrieval Systems, *Information Processing and Management,* 12 (2), pp. 147-153

[BRES78]
Bresnan J (1978). A Realistic Transformational Grammar, in Halle M, Bresnan J and Miller GA (Eds), *Linguistic Theory and Psychological Reality,* Cambridge, Mass.: MIT Press

[BRES82]
Bresnan J (1982). *The Mental Representation of Grammatical Relations,* Cambridge, Mass.: MIT Press

[BUND83]
Bundy A (Ed) (1983). *Proceedings 8th IJCAI Conference,* Karlsruhe, West Germany

[BURG75]
Burger JF, Leal A and Shoshani A (1975). Semantic-Based Parsing and a Natural-Language Interface to Interactive Data Management, *American Journal of Computational Linguistics,* Microfiche 32, pp. 58-71

[BURG80]
Burger JF (1980). Semantic Database Mapping in EUFID, *Proceedings 1980 ACM-SIGMOD Conference,* Santa Monica, Ca

[BURG82]
Burger JF and Templeton M (1982). Recommendations for an Internal Input Language for the Knowledge-Based Systems, System Development Corporation internal paper N-(L-)24890/021/00, January 5

[BUSE84]
Busemann S (1984). Surface Transformations During the Generation of Written German Sentences, in [BOLC84]

[BUSE85]
Busemann S, Hoeppner W, Christaller T, Morik K (1985). Representing and Processing Copula and Full-Verb Sentences in HAM-ANS, in Stoyan H (ed), *GWAI-85,* Berlin: Springer, pp. 187-196

[CADI76]
Cadiou J-M (1976). On Semantic Issues in the Relational Model of Data, *Proceedings 5th Symposium on Mathematical Foundations of Computer Science,* pp. 23-38
[CARB81]
Carbonell JG (1981). *Subjective Understanding: Computer Models of Belief Systems,* Ann Arbor, Mich
[CARB83a]
Carbonell JG (1983). Learning by Analogy: Formulating and Generalizing Plans from Past Experience, in Michalski RS, Carbonell JG and Mitchell TM (Eds), *Machine Learning,* pp. 137-161, Palo Alto, Ca: Tioga
[CARB83b]
Carbonell JG and Michalski RS (1983). Machine Learning: A Historical and Methodological Analysis, *The AI Magazine,* 4 (3), pp. 69-73
[CHAF76]
Chafe WL (1976). Givenness, Contrastiveness, Definiteness, Subjects, Topics, and Point of View, in Li, C N (Ed), *Subject and Topic,* New York: Academic Press
[CHAP73]
Chapanis A (1973). The Communication of Factual Information through Various Channels, *Information Storage and Retrieval,* 9, pp. 215-231
[CHOM65]
Chomsky N (1965). *Aspects of the Theory of Syntax,* Cambridge
[CHRI82]
Christaller T (1982). Konsistenzüberprüfungen bei sich ändernden Wissensbasen, in Wahlster, W. (Ed), *GWAI-82,* pp. 63-71, Heidelberg: Springer
[CODD70]
Codd EF (1970). A Relational Model of Data for Large Shared Data Banks, *Communications of the ACM,* 13 (6), pp. 377-387
[CODD71a]
Codd EF (1971). A Data Base Sublanguage Founded on the Relational Calculus, *Proceedings ACM-SIGFIDET Workshop on Data Description, Access and Control,* pp. 35-68
[CODD71b]
Codd EF (1971). Further Normalization of the Data Base Relational Model, in Rustin R (Ed), *Data Base Systems,* pp. 33-64, Englewood Cliffs, NJ: Prentice-Hall
[CODD74]
Codd EF (1974). Seven Steps to Rendezvous with the Casual User, in Klimbie JW and Koffeman KI (Eds), *Data Base Management,* pp. 179-200, Amsterdam: North-Holland
[CODD78]
Codd EF, Arnold RS, Cadiou J-M, Chang CL, Roussopoulos N (1978). RENDEZVOUS Version 1: An Experimental English-Language Query Formulation System for Casual Users of Relational Data Bases, IBM Research Report RJ2144, San Jose, Ca
[COHE82]
Cohen PR, Perrault CR and Allen JF (1982). Beyond Question Answering, in Lehnert WG and Ringle MH (Eds), *Strategies for Natural Language Processing,* pp. 245-274, Hillsdale, NJ: Lawrence Erlbaum
[CULL80]
Cullinane Corporation (1980). IQS Summary Description, May
[DAME80]
Damerau FJ (1980). The Transformational Question Answering (TQA) System: Description, Operating Experience, and Implications, IBM Research Report RC 8287, Yorktown Heights, NY
[DAME81]
Damerau FJ (1981). Operating Statistics for the Transformational Question Answering System, *American Journal of Computational Linguistics,* 7 (1), pp. 30-42
[DAME83]
Damerau FJ (1983). Shape Descriptor for Unidentified Words in a Natural Language Processing System, *IBM Technical Disclosure Bulletin,* 26, pp. 2648-2649

[DATE77]
Date CJ (1977). *An Introduction to Database Systems,* 2nd ed., Menlo Park, Ca: Addison-Wesley
[DATE81]
Date CJ (1981). *An Introduction to Database Systems,* 3rd ed, Reading, Mass.: Addison-Wesley
[DAVI80]
Davis R (1980). Meta-Rules: Reasoning about Control, *Artificial Intelligence,* 5, pp. 179-222
[DEFU84]
Defude B (1984). Knowledge Based Systems Versus Thesaurus: An Architecture Problem about Expert Systems Design, in van Rijsbergen CJ (Ed), *Research and Developments in Information Retrieval,* pp. 267-280, Cambridge, UK: Cambridge University Press
[DELL77]
Dell'Orco P, Spadavecchia VN and King M (1977). Using Knowledge of a Data Base World in Interpreting Natural Language Queries, *IFIP 77 Congress Proceedings,* pp. 139-144
[DONE78]
Donelson W (1978). Spatial Management of Data, *Proceedings ACM-SIGGRAPH Conference,* Atlanta, Ga
[DOSZ79]
Doszkocs TE and Rapp BA (1979). Searching MEDLINE in English: A Prototype User Interface with Natural Language Query, Ranked Output, and Relevance Feedback, *Proceedings 42nd ASIS Annual Meeting,* 16, pp. 131-137
[DRES76]
Dresher BE and Hornstein N (1976). On some Supposed Contributions of Artificial Intelligence to the Scientific Study of Language, *Cognition,* 4, pp. 321-398
[EDP82]
Query Systems for End Users, *EDP Analyzer,* 20 (9), September 1982
[EMBL81]
Embley DW and Nagy G (1981). Behavioral Aspects of Text Editors, *ACM Computing Surveys,* 13 (1), pp. 33-70
[ERMA81]
Erman LD, London PE and Fickas SF (1981). The Design and an Example of Use of HEARSAY-III, *Proceedings 7th IJCAI Conference,* Vancouver, BC, pp. 409-515
[FINN79]
Finnin T, Goodman B and Tennant H (1979). JETS: Achieving Completeness through Coverage and Closure, Working Paper, University of Illinois, Champaign/Urbana
[FISC82]
Fischer HG (Ed) (1982). *Information Retrieval und natürliche Sprache. Integrierte Verarbeitung von Daten und Texten im Modell CONDOR,* Munich, West Germany
[FLIE83]
Fliegner M (1983). Überlegungen zur automatischen Schreibfehlerkorrektur für ein KI-System, Memo GEN-18, Universität Hamburg, Forschungsstelle für Informationswissenschaft und Künstliche Intelligenz
[GAZD81]
Gazdar G, Pullum GK and Sag I (1981). Auxiliaries and Related Phenomena in a Restrictive Theory of Grammar, *Language,* 58, pp. 591-638
[GHOS77]
Ghosh SP (1977). *Data Base Organization for Data Management,* New York, NY: Academic Press
[GNAN80]
Gnanamgari S, (1980). Providing Automatic Graphic Displays through Defaults, *Proceedings Third National Conference of the Canadian Society for Computational Studies of Intelligence*
[GOLD83]
Goldberg A and Robson D (1983). *Smalltalk-80: The Language and Its Implementation,* Reading, Mass.: Addison-Wesley

[GREE63]
Green BF, Wolf AK, Chomsky C and Laughery K (1963). Baseball: An Automatic Question Answerer, in Feigenbaum E and Feldman J (Eds), *Computers and Thought,* pp. 207-233, New York, NY: McGraw-Hill

[GREE78]
Greenblatt D and Waxman J (1978). A Study of Three Database Query Languages, in Shneiderman B (Ed), *Databases: Improving Usability and Responsiveness,* pp. 77-97, New York: Springer

[GREE81]
Greenfeld NR (1981). Jericho: A Professional's Personal Computer System, *Proceedings 8th Intl. Symp. Computer Architecture,* Rochester, Maine

[GROS78]
Grosz BJ (1978). Focusing in Dialog, in Waltz DL (Ed), *TINLAP-2: Theoretical Issues in Natural Language Processing-2,* pp. 96-103, Urbana-Champaign

[GROS80a]
Grosz BJ and Hendrix G (1980). Focusing and Description in Natural Language Dialogues, in Joshi AK et al. (Eds), *Elements of Discourse Understanding,* Cambridge, UK: Cambridge University Press

[GROS80b]
Grosz BJ and Hendrix G (1980). A Computational Perspective on Indefinite Reference, Artificial Intelligence Center Technical Note No 181, Menlo Park, Ca.: SRI International

[GROS83a]
Grosz BJ (1983). TEAM: A Transportable Natural-language Interface System, *Proceedings Applied Natural Language Conference,* Santa Monica, Ca

[GROS83b]
Grosz B and Stickel M (1983). Research on Interactive Acquisition and Use of Knowledge, SRI Project 1894, Final Report, Menlo Park, Ca.: SRI International

[GUID82]
Guida G and Tasso C (1982). NLI: A Robust Interface for Natural Language Person-Machine Communication, *International Journal of Man-Machine Studies,* 17, pp. 417-433

[GUID83a]
Guida G and Tasso C (1983). IR-NLI: An Expert Natural Language Interface to Online Data Bases, *Proceedings ACL Conference on Applied Natural Language Processing,* pp. 31-38, Santa Monica, Ca

[GUID83b]
Guida G and Tasso C (1983). An Expert Intermediary System for Interactive Document Retrieval, *Automatica,* 19 (6), pp. 759-766

[GUID84]
Guida G and Tasso C (1984). A New Approach to the Design of Expert System Architectures, in Plander I (Ed), *Proceedings 3rd Intl Conf on Artificial Intelligence and Information - Control Systems of Robots,* pp. 405-423, Amsterdam: North-Holland

[HABE79]
Habel C, Rollinger CR, Schmidt A and Schneider H-J (1979). Ein logik-orientierter Ansatz zum automatischen Text-verstehen, Report No 1/79, Technische Universität Berlin, Semantic Network Project

[HAHN80]
von Hahn W, Hoeppner W, Jameson A and Wahlster W (1980). The Anatomy of the Natural Language Dialogue System HAM-RPM, in [BOLC80], pp. 119-253

[HAHN83]
von Hahn W (1983). The Contribution of Artificial Intelligence to the Human Factors of Application Software, in Blaser A, Zoeppritz M (Eds), *Enduser Systems and their Human Factors,* pp. 128-138, Heidelberg: Springer

[HARR77a]
Harris LR (1977). Natural Language Data Base Query: Using the Data Base Itself as the Definition of World Knowledge and as an Extension of the Dictionary, Technical Report TR 77-2, Dartmouth College, Department of Mathematics, Hannover

[HARR77b]
Harris LR (1977). ROBOT: A High Performance Natural Language Data Base Query System, *Proceedings 5th IJCAI Conference,* pp. 903-904

[HARR77c]
Harris LR (1977). User Oriented Data Base Query with the ROBOT Natural Language System, *Proceedings 3rd VLDB Conference,* Tokyo, pp. 303-311

[HARR78]
Harris LR (1978). The ROBOT System: Natural Language Processing Applied to Data Base Query, *Proceedings ACM 78 Annual Conference*

[HARR79]
Harris LR (1979). Experience with ROBOT in 12 Commercial Natural Language Data Base Query Applications, *Proceedings 6th IJCAI Conference,* pp. 365-368

[HAYE79]
Hayes PJ and Reddy R (1979). An Anatomy of Graceful Interaction in Spoken and Written Man-Machine Communication, CMU-CS-79-144, Carnegie-Mellon University, Department of Computer Science, Pittsburgh, PA

[HAYE81]
Hayes PJ and Carbonell JG (1981). Multi-Strategy Parsing and its Role in Robust Man-Machine Communication, Report CMU-CS-81-118, Carnegie-Mellon University, Pittsburgh

[HAYE83]
Hayes-Roth F, Waterman DA and Lenat DB (Eds) (1983). *Building Expert Systems,* Reading, Mass: Addison-Wesley

[HEND77]
Hendrix GG (1977). Human Engineering for Applied Natural Language Processing, *Proceedings 5th IJCAI,* pp. 183-191

[HEND78a]
Hendrix GG (1978). A Natural Language Interface Facility and its Application to a IIASA Data Base, in Rahmstorf G and Ferguson M (Eds), *Proceedings Workshop on Natural Language for Interaction with Data Bases,* Laxenburg, Austria, pp. 87-94

[HEND78b]
Hendrix GG (1978). Semantic Aspects of Translation, in Walker DE (Ed), *Understanding Spoken Language,* pp. 193-228, New York

[HEND78c]
Hendrix GG, Sacerdoti ED, Sagalowicz D and Slocum J (1978). Developing a Natural Language Interface to Complex Data, *ACM Transactions on Database Systems,* 3 (2), pp. 105-147

[HERK80]
Herkner W (Ed) (1980). *Attribution - Psychologie der Kausalität,* Bonn, Stuttgart, Wien

[HERO80]
Herot CF and Wilson GA (1980). Semantics Versus Graphics - To Show or Not to Show, Technical Report CCA-80-09, Cambridge, Mass.: Computer Corporation of America

[HOBB79]
Hobbs J and Robinson J (1979). "Why ask", *Discourse Processes,* 2 (4)

[HOEN82]
Hoenkamp E, Kempen G (1982). Incremental Sentence Generation: Implications for the Structure of a Syntactic Processor, in Horecky J (Ed), *Proceedings COLING 82,* pp. 151-156, Amsterdam, New York, Oxford

[HOEP80]
Hoeppner W (1980). Repräsentationsstrukturen und Inferenzen für zusammengesetzte Objekte, in Rollinger C-R, Schneider H-J (Eds), *Inferenzen in natürlichsprachlichen Systemen der KI,* pp. 151-172, Berlin

[HOEP82]
Hoeppner W (1982). ATN-Steuerung durch Kasusrahmen, in Wahlster W (Ed), *GWAI-82,* pp. 213-226, Heidelberg: Springer

[HOEP83]
Hoeppner W, Christaller T, Marburger H, Morik K, Nebel B, O'Leary M and Wahlster W (1983). Beyond Domain-Independence: Experience with the Development of a German Language Access System to Highly Diverse Background Systems, in [BUND83], pp. 588-594

[HOEP84]
Hoeppner W, Busemann S, Christaller T, Marburger H, Morik K, Nebel B (1984). Commented Terminal Sessions with a Natural Language System, Research Unit for Information Science and Artificial Intelligence, Memo ANS-23, University of Hamburg

[HONE76]
Honeywell (1976). WWMCCS: World Wide Data Management System User's Guide, Honeywell DB97 Rev 3, April

[HUSS82]
Hussmann M and Genzmann H (1982). Performanz-orientiertes Parsing - Ansätze zur robusten Analyse natürlicher Sprache, Memo GEN-5, Universität Hamburg, Forschungsstelle für Informationswissenschaft und Künstliche Intelligenz

[IBM76]
IBM (1976). MIS/370 Anwendung. IV Informations-Systeme

[IBM81]
IBM Corporation (1981). SQL/Data System Concepts and Facilities, Endicott, New York: IBM Corporation

[JACK77]
Jackendoff R (1977). *X-Bar Syntax: A Study of Phrase Structure,* Cambridge, Mass: MIT Press

[JAME82a]
Jameson A (1982). Documentation for Three HAM-ANS Components: Ellipsis, NORMALIZE and NORMALIZE-1, Memo ANS-4, Universität Hamburg, Forschungsstelle für Informationswissenschaft und Künstliche Intelligenz

[JAME82b]
Jameson A and Wahlster W (1982). User Modeling in Anaphora Generation: Ellipsis and Definite Description, *Proceedings ECAI-82,* Orsay, France, pp. 222-227

[JAME83]
Jameson A (1983). Impression Monitoring in Evaluation-Oriented Dialog. The Role of the Listener's Assumed Expectations and Values in the Generation of Informative Statements, in [BUND83], pp. 616-620

[JANA79]
Janas JM (1979). Towards more Informative User Interfaces, *Proceedings 5th VLDB Conference,* Rio de Janeiro, pp. 17-23

[JANA82]
Janas JM (1982). Natürlichsprachliche Schnittstellen zu relationalen Datenbanken: Ein semantisch orientierter Ansatz, PhD Dissertation, Hochschule der Bundeswehr München, Neubiberg

[JARK83]
Jarke M (1983). Zur Beurteilung natürlichsprachlicher Endbenutzerschnittstellen von Datenbanken, in Schmidt JW (Ed), *Sprachen für Datenbanken,* pp. 42-60, Heidelberg: Springer

[JARK85a]
Jarke M, Turner JA, Stohr EA, Vassiliou Y, White NH and Michielsen K (1985). A Field Evaluation of Natural Language for Data Retireval, *IEEE Transactions on Software Engineering,* SE-11 (1), pp. 97-114

[JARK85b]
Jarke M and Vassiliou Y (1985). A Framework for Choosing a Database Query Language, *ACM Computing Surveys,* 17 (3), pp. 213-240
[JARK86]
Jarke M (Ed) (1986). *Managers, Micros, and Mainframes: Integrating Systems for End Users,* London: John Wiley
[JOSH80]
Joshi AK and Levy LS (1980). Phrase Structure Trees Bear more Fruit than You would Have Thought, University of Pennsylvania, Philadelphia, Pa
[KAPL79]
Kaplan SJ (1979). Cooperative Responses from a Portable Natural Language Data Base Query System, Ph D dissertation, University of Pennsylvania, Philadelphia, Pa
[KAPL83]
Kaplan SJ (1983). Cooperative Responses from a Portable Natural Language Database Query System, in Brady M and Berwick KC (Eds), *Computational Models of Discourse,* pp. 167-208, Cambridge, Mass: MIT Press
[KAY83]
Kay M (1983). Unification Grammar (Ms)
[KAYG76]
Kay A and Goldberg A (1976). Personal Dynamic Media, Technical Report SSL 76-1, Xerox Palo Alto Research Center, Palo Alto, Ca
[KELL71]
Kellogg CH, Burger JF, Diller T and Fogt K (1971). The CONVERSE Natural Language Data Management System: Current Status and Plans, *Proceedings ACM Symposium on Information Storage and Retrieval,* pp. 33-46, College Park, MD: University of Maryland
[KETT81]
Kettler W, Schmidt A and Zoeppritz M (1981). Erfahrungen mit zwei natürlichsprachlichen Abfragesystemen, TR 81.01. 001, IBM Heidelberg Scientific Center
[KNUT68]
Knuth DE (1968). Semantics of Context-Free Languages, *Mathematical Systems Theory,* II, pp. 127-145
[KOLV79]
Kolvenbach M, Loetscher A, Dutz HD (Eds) (1979). *Künstliche Intelligenz und natürliche Sprache. Sprachverstehen und Problemlösungen mit dem Computer,* Tübingen
[KONO79]
Konolige K (1979). A Framework for a Portable Natural-Language Interface to Large Data Bases, Technical Note 197, Stanford Research Institute, Artificial Intelligence Center
[KRAU80a]
Krause J (1980). Natural Language Access to Information Systems: An Evaluation Study of its Acceptance by End Users, *Information Systems,* 4, pp. 297-318
[KRAU80b]
Krause J and Lehmann H (1980). The User Specialty Language. A Natural Language Based Information System and its Evaluation, in Krallmann D (Ed), *Dialogsysteme und Textverarbeitung,* pp. 127-146, Essen
[KRAU82]
Krause J (1982). *Mensch-Maschine-Kommunikation in natürlicher Sprache,* Tübingen: Niemeyer
[KRAU83a]
Krause J (1983). Praxisorientierte natürlichsprachliche Frage-Antwort-Systeme: Zur Entwicklung vor allem in der Bundesrepublik Deutschland, *Nachrichten für Dokumentation,* 34 (4/5), pp. 188-194
[KRAU83b]
Krause J (1983). Linguistic Components in (Office) Information Systems and a General Evaluation Strategy for Automatic Indexing, *Journal of Information and Optimization Sciences,* 87

[KRAU83c]
Krause J, Schneider C, Spettel G and Wormser-Hacker C (1983). *EVAL: Zur Evaluierung informationslinguistischer Komponenten von Informationssystemen,* Regensburg

[LANC79]
Lancaster FW (1979). *Information Retrieval Systems,* New York: John Wiley

[LAND77]
Landsbergen SP and Scha RJH (1977). An Outline of the Question-Answering System PHLIQA1, Reprints from the Workshop on Logic and Data Bases, pp. XI.1-XI.16, Toulouse

[LANG78]
Langacker RW (1978). The Form and Meaning of the English Auxiliary, *Language,* 54, pp. 853-882

[LEFA77]
Lefaivre RA (1977). FUZZY Reference Manual, New Brunswick, NJ: Rutgers University

[LEHM78a]
Lehmann H (1978). Interpretation of Natural Language in an Information System, *IBM Journal of Research and Development,* 22 (5), pp. 560-572

[LEHM78b]
Lehmann H, Ott N and Zoeppritz M (1978). User Experiments with Natural Language for Database Access, *Proceedings 7th International Conference on Computational Linguistics,* Bergen

[LEHM79]
Lehmann H and Blaser A (1979). Query Languages in Database Systems, TR 79.07.004, IBM Heidelberg Scientific Center

[LEVE82]
Levelt W (1982). Linearization in Describing Spatial Networks, in Peters S and Saarinen E (Eds), *Processes, Beliefs, and Questions. Essays on Formal Semantics of Natural Language and Natural Language Processing,* pp. 199-220, Dordrecht

[LIND63]
Lindsay RK (1963). Inferential Memory as the Basis of Machines which Understand Natural Language, in Feigenbaum EA and Feldman J (Eds), *Computers and Thought,* New York: McGraw-Hill

[MALH75]
Malhotra A (1975). Design Criteria for a Knowledge-Based English Language System for Management: An Experimental Analysis, PhD dissertation, Massachusetts Institute of Technology, Cambridge, Mass

[MALK82]
Malkovsky M (1982). TULIPS-2 Natural Language Learning System, *Proceedings Coling 82,* Prague

[MARB83a]
Marburger H and Nebel B (1983). Natürlichsprachlicher Datenbankzugang mit HAM-ANS: Syntaktische Korrespondenz, natürlichsprachliche Quantifizierung und semantisches Modell des Diskursbereichs, in Schmidt JW (Ed), *Sprachen für Datenbanken,* pp. 26-41, Berlin: Springer

[MARB83b]
Marburger H and Wahlster W (1983). Case Role Filling as a Side Effect of Visual Search, *Proceedings First EACL Meeting,* Pisa, pp. 188-195

[MARB85]
Marburger H (1985). Kooperativität in natürlichsprachlichen Zugangssystemen, in Brauer W, Radig B (eds), *Wissensbasierte Systeme. GI-Kongreß 1985,* Berlin: Springer, pp. 135-144

[MARC81a]
Marcus RS (1981). A Translating Computer Interface for End-User Operation of Heterogeneous Retrieval Systems: I. Design, II. Evaluations, *Journal of the American Society for Information Science,* 32, pp. 287-317

[MARC81b]
Marcus RS (1981). An Automated Expert Assistant for Information Retrieval in the Information Community: An Alliance for Progress, *Proceedings 44th ASIS Annual Meeting,* pp. 270-273

[MARS80]
Marslen-Wilson WD and Tyler LK (1980). The Temporal Structure of Spoken Language Understanding, *Cognition,* 8, pp. 1-71

[MART83]
Martin PA, Appelt DA and Pereira FC (1983). Transportability and Generality in a Natural-Language Interface System, in [BUND83], pp. 573-581

[MCKE82]
McKeown K (1982). Generating Natural Language Text in Response to Questions about Database Structure, Ph D dissertation, University of Pennsylavania, Philadelphia, Pa

[MEAD81]
Meadow CT and Cochrane PA (1981). *Basics of Online Searching,* New York: John Wiley

[MEEH79]
Meehan JR (1979). The New UCI-LISP Manual, Hillsdale

[MICH83]
Michalski RS (1983). A Theory and Methodology of Inductive Learning, in Michalski RS, Carbonell JG and Mitchell TM (Eds), *Machine Learning,* pp. 83-134, Palo Alto, Ca: Tioga

[MILL81]
Miller LA (1981). Natural Language Programming: Styles, Strategies, and Contrasts, *IBM Systems Journal,* 20 (2), pp. 184-215

[MORA81]
Moran T (1981). An Applied Psychology of the User, *ACM Computing Surveys,* 13 (1), pp. 1-12

[MORI81]
Morik K (1981). Verarbeitung von externer und interner Situation in Überzeugungssystemen, in Siekmann J (Ed), *GWAI-81,* pp. 287-296, Berlin: Springer

[MORI82]
Morik K (1982). Differenzstudie zu früheren sprachverarbeitenden Systemen in der Bundesrepublik Deutschland, HAM-ANS Report 6, Forschungsstelle für Informationswissenschaft und Künstliche Intelligenz, Universität Hamburg

[MORI83a]
Morik K (1983). Wertäußerungen und Repräsentation von Bewertungen, Memo ANS-14, Universität Hamburg, Forschungsstelle für Informationswissenschaft und Künstliche Intelligenz

[MORI83b]
Morik K (1983). Demands and Requirements for Natural Language Systems - Results of an Inquiry, in [BUND83], pp. 647-649

[MORI83c]
Morik K and Rollinger C-R (1983). Partnermodellierung im Evidenzraum, in Neumann B (Ed), *GWAI-83,* pp. 158-168, Berlin: Springer

[MORI85]
Morik K (1985). User Modeling, Dialog Structure and Dialog Strategy in HAM-ANS, *Proceedings 2nd EACL Conference,* Geneva, pp. 268-273

[MYLO76]
Mylopoulos J, Borgida A, Cohen P, Roussopoulos N, Tsotsos J and Wong H (1976). TORUS: A Step Towards Bridging the Gap between Data Bases and the Casual User, *Information Systems,* 2 (1), pp. 49-64

[NEUM82]
Neumann B (1982). Towards Natural Language Description of Real-World Image Sequences, in Nehmer J (Ed), *GI - 12. Jahrestagung,* pp. 349-358, Berlin: Springer

[NIEVE83]
Nievergelt J (1983). Die Gestaltung der Mensch-Maschine-Schnittstelle, in Schmidt JW (Ed), *Sprachen für Datenbanken,* pp. 1-10, Berlin: Springer

[O'LE83]
O'Leary M (1983). PEVAL: Towards an Interface Between the HAM-ANS Core System and PASCAL/R Databases, Memo ANS-15, Universität Hamburg, Forschungsstelle für Informationswissenschaft und Künstliche Intelligenz

[OLNE78]
Olney J (1978). Enabling EUFID to Handle Negative Expressions, SDC SP-3996 Systems Development Corporation

[OSI79]
Operating Systems, Inc. (1979). An Assessment of Natural Language Interfaces for Command and Control Database Query, Logicon/OSI Division report for WWMCCS System Engineering, OSI Report R79-026

[OTT79a]
Ott N (1979). Bericht über die KFG-Studie, IBM Heidelberg Scientific Center TN 79.03

[OTT79b]
Ott N (1979). Das experimentelle, auf natürlicher Sprache basierende Informationssystem USL, *Nachrichten für Dokumentation,* 30 (3), pp. 129-140

[OTTZ80]
Ott N and Zoeppritz M (1979). USL - An Experimental Information System Based on Natural Language, in [BOLC80], pp. 256-282

[PAXT77]
Paxton WH (1977). A Framework for Speech Understanding, Ph D thesis, Stanford University, Stanford, Ca

[PETR73]
Petrick SR (1973). Transformational Analysis, in Rustin R (Ed), *Natural Language Processing,* pp. 27-41, New York: Algorithmics Press

[PETR76]
Petrick SR (1976). On Natural Language Based Computer Systems, *IBM Journal of Research and Development,* 20 (4), pp. 314-325

[PIRO78]
Pirotte A (1978). Linguistic Aspects of High-Level Relational Languages, in *Data Base Technology, 2 (Infotech State of the Art Report),* pp. 271-300, Maidenhead, UK

[PLAT76]
Plath WJ (1976). REQUEST: A Natural Language Question-Answering System, *IBM Journal of Research and Development,* 20 (4), pp. 326-365

[POLL81]
Pollitt AS (1981). An Expert System as an Online Search Intermediary, *Proceedings 5th Intl. Online Information Meeting,* London, pp. 25-32

[POLL82]
Pollitt AS (1982). A Search Statement Generator for Cancer Therapy Related Information Retrieval, *Proceedings 6th Intl. Online Information Meeting,* London, pp. 405-413

[REIC84]
Reichman-Adar R (1984). Extended Person-Machine Interface, *Artificial Intelligence,* 22, pp. 157-218

[REIS77]
Reisner P (1977). Use of Psychological Experiments as an Aid to the Development of a Query Language, *IEEE Transactions on Software Engineering,* SE-3 (3), pp. 218-229

[REIS81]
Reisner P (1981). Human Factors Studies of Database Query Languages: A Survey and Assessment, *ACM Computing Surveys,* 13 (1), pp. 13-31

[REIT83]
Reiter R (Chairperson), Gallaire H, King JJ, Mylopoulos J and Webber BL (1983). A Panel on AI and Databases, in [BUND83], pp. 1199-1206

[RICH79]
Rich E (1979). Building and Exploiting User Models, Report CMU-CS-79-119, Carnegie-Mellon University, Pittsburgh, Pa
[ROBI70]
Robinson JR (1970). Dependency Structures and Transformational Rules, *Language,* 46, pp. 259-285
[ROBI73]
Robinson JR (1973). An Inverse Transformational Lexicon, in Rustin R (Ed), *Natural Language Processing,* pp. 43-60, New York: Algorithmics Press
[ROSS69]
Ross JR (1969). Adjectives as Noun Phrases, in Reibel DA and Schane SA (Eds), *Modern Studies in English,* Englewood Cliffs, NJ: Prentice Hall
[SALT83]
Salton G and McGill MJ (1983). *Introduction to Modern Information Retrieval,* New York: McGraw-Hill
[SCHA77]
Scha RJH (1977). Phillips Question-Answering System PHLIQA1, in [WALT77]
[SCHM76]
Schmidt SJ (1976). *Texttheorie,* Munich, West Germany
[SCHM80]
Schmidt JW and Mall M (1980). PASCAL/R Report, Report IFI-HH-B-66/80, Universität Hamburg
[SCHN84]
Schneider M (1984). Ergonomic Considerations in the Design of Control Languages, in Vassiliou Y (Ed), *Human Factors and Interactive Computer Systems,* pp. 141-161, Norwood, NJ: Ablex
[SCHU79a]
Schuetz A and Luckmann T (1979). *Strukturen der Lebenswelt,* Frankfurt
[SCHU79b]
Schuetz F (1979). Noten am KFG. Zufall oder Notwendigkeit, in Elternvereinigung des KFG Mannheim (Eds), *Karl-Friedrich-Gymnasium Mannheim. Jahresbericht,* pp. 39-87
[SEAR77]
Searle JR (1977). A Classification of Illocutionary Acts, *Proceedings Texas Conference on Performatives, Presuppositions and Implicatures,* Arlington, Va
[SEAR78]
Searle JR (1978). Intentionality and the Use of Language, *Studies in the Linguistics Sciences,* 8 (2), pp. 149-162, Urbana: University of Illinois
[SHNE80]
Shneiderman B (1980). *Software Psychology,* Winthrop
[SHWA84]
Shwartz SP (1984). Natural Language Processing in the Commercial World, in Reitman W (Ed), *Artificial Intelligence Applications for Business,* pp. 235-248, Norwood, NJ: Ablex
[SIMM65]
Simmons RF (1965). Answering English Questions by Computer - a Survey, *Communications of the ACM,* 8 (1), pp. 53-70
[SMAL77]
Small D and Weldon LJ (1977). The Efficiency of Retrieving Information from Computers using Natural and Structured Query Languages, Report SAI-78-655-WA, Science Applications, September
[STOH82]
Stohr EA, Turner JA, Vassiliou Y and White NH (1982). Research in Natural Language Systems, *Proceedings 15th Annual Hawaii Conference on System Sciences,* Honolulu, Hw
[STON76]
Stonebraker M, Wong E, Kreps P and Held G (1976). The Design and Implementation of INGRES, *ACM Transactions on Database Systems,* 1 (3), pp. 189-222

[SUTH63]
Sutherland IC (1963). Sketchpad: A Man-Machine Graphical Communications System, Technical Report LL-TR-296, Lincoln Labs, MIT, Cambridge, Mass.
[SYST83]
Systems-Software, August 1983
[TEMP79]
Templeton MP (1979). EUFID: A Friendly and Flexible Frontend for Data Management Systems, *Proceedings National Conference of the Association for Computational Linguistics*
[TEMP80]
Templeton MP (1980). A Natural Language User Interface, *Proceedings "Pathways to System Integrity"*, Washington, D.C., Chapter of the ACM
[TENN79]
Tennant HR (1979). Evaluation of Natural Language Processors, Ph D dissertation, University of Illinois, Urbana
[TESN76]
Tesniere L (1976). *Elements de Syntaxe Structurale*, 2nd Ed, Paris: Editions Klincksieck
[THOM69]
Thompson FB, Lockemann PC, Dostert BH and Deverill R (1969). REL: A Rapidly Extensible Language System, *Proceedings 24th ACM National Conference*, pp. 399-417
[THOM83]
Thompson B and Thompson F (1983). Introducing Ask, A Simple Knowledgeable System, *Proceedings Conference on Applied Natural Language Processing*, pp. 17-24
[TODD76]
Todd SJP (1976). The Peterlee Relational Test Vehicle, *IBM Systems Journal*, 15 (4), pp. 285-308
[TURN84]
Turner JA, Jarke M, Stohr EA, Vassiliou Y and White NH (1984). Using Restricted Natural Language for Data Retrieval - A Plan for Field Evaluation, in Vassiliou Y (Ed), *Human Factors and Interactive Computer Systems*, pp. 163-190. Norwood, NJ: Ablex
[VASS83a]
Vassiliou Y, Jarke M, Stohr EA, Turner JA and White NH (1983). Natural Language for Database Queries: A Laboratory Study, *MIS Quarterly*, 7 (4), pp. 47-61
[VASS83b]
Vassiliou Y, Jarke M, Stohr EA, Turner JA and White NH (1983). Application Development for Natural Language Query Systems, *Proceedings IEEE Workshop on Languages for Automation*, Chicago, pp. 288-293
[VASS84]
Vassiliou Y and Jarke M (1984). Query Languages - A Taxonomy, in Vassiliou Y (Ed), *Human Factors and Interactive Computer Systems*, pp. 47-82. Norwood, NJ: Ablex
[WAHL81]
Wahlster W (1981). *Natürlichsprachliche Argumentation in Dialogsystemen. KI-Verfahren zur Rekonstruktion und Erklärung approximativer Inferenzprozesse*, Berlin: Springer
[WAHL82]
Wahlster W (1982). Natürlichsprachliche Systeme. Eine Einführung in die sprachorientierte KI-Forschung, in Bibel W and Siekmann JH (Eds), *Künstliche Intelligenz. Frühjahrsschule Teisendorf*, pp. 203-284, Berlin: Springer
[WAHL83]
Wahlster W, Marburger H, Jameson A and Busemann S (1983). Over-Answering Yes-No Questions: Extended Responses in a NL Interface to a Vision System, in [BUND83], pp. 643-646
[WALT76]
Waltz DL, Conrad F, Finin T, Goodman B, Green F, Hadden G (1976). The PLANES System: Natural Language Access to a Large Data Base, Report T-34, University of Illinois, Coordinated Science Laboratory, Urbana
[WALT77]
Waltz DL (Ed) (1977). Natural Language Interfaces, *ACM SIGART Newsletter*, 61

[WALT78]
Waltz DL (1978). An English Language Question Answering System for a Large Relational Database, *Communications of the ACM,* 21 (7), pp. 526-539
[WALT83]
Waltz DL (1983). Artificial Intelligence: An Assessment of the State-of-the-Art and Recommendations for Future Directions, *The AI Magazine,* 4 (3), pp. 55-67
[WASO78]
Wasow T (1978). Remarks on Processing, Constraints, and the Lexicon, in Waltz DL (Ed), *TINLAP-2: Theoretical Issues in Natural Language Processing-2,* Urbana: University of Illinois
[WATE78]
Waterman DA and Hayes-Roth F (Eds) (1978). *Pattern-Directed Inference Systems,* New York, NY: Academic Press
[WEIN81]
Weinreb D and Moon D (1981). Lisp Machine Manual, Cambridge, MA: Massachusetts Institute of Technology, Artificial Intelligence Laboratory
[WEIS80]
Weischedel RM and Black JE (1980). Responding Intelligently to Unparsable Inputs, *American Journal of Computational Linguistics,* 6 (2), pp. 97-109
[WELT81]
Welty C and Stemple DW (1981). Human Factors Comparison of a Procedural and a Non-Procedural Query Language, *ACM Transactions on Database Systems,* 6 (4), pp. 629-649
[WILE83]
Wilensky R (1983). *Planning and Understanding,* Reading, MA: Addison-Wesley
[WOOD70]
Woods WA (1970). Transition Network Grammars for Natural Language Analysis, *Communications of the ACM,* 13 (10), pp. 591-606
[WOOD72]
Woods WA, Kaplan RM and Nash-Webber B (1972). The Lunar Sciences Natural Language Information System: Final Report, BBN Report 2378, Bolt Beranek and Newman Inc., Cambridge, MA
[WOOD77a]
Woods WA (1977). Lunar Rocks in Natural English: Explorations in Natural Language Question Answering, in Zampolli (Ed), *Linguistic Structures Processing,* Amsterdam: North Holland
[WOOD77b]
Woods WA (1977). A Personal View of Natural Language Understanding, in [WALT77], pp. 17-20
[YONK78]
Yonke MD (1978). A Lisp Machine Should Only be Thought of as an Expensive Terminal, *Proceedings 11th Annual Microprogramming Workshop,* Pacific Grove, Ca
[ZOEP83]
Zoeppritz M (1983). Human Factors of a 'Natural Language' Enduser System, in Blaser A and Zoeppritz M (Eds), *Enduser Systems and their Human Factors,* pp. 62-93, Heidelberg: Springer
[ZOEP84a]
Zoeppritz M (1984). *Syntax for German in the User Specialty Languages System,* Tübingen
[ZOEP84b]
Zoeppritz M (1984). Datenabfrage in natürlicher Sprache: Diskussion von Eigenschaften der natürlichen Sprache aufgrund von Beobachtungsdaten, in Ochl, H (Ed), *Online '84 Proceedings,* Berlin.
[ZOLT82]
Zoltan, E., Weeks, G. and Ford, WR (1982). Natural Language Communication with Computers: A Comparison of Voice and Keyboard Input, in Johannsen, G and Rijsdorp, JE (Eds), *Analysis, Design, and Evaluation of Man-Machine Systems,* pp. 27-28, Baden-Baden