PUC

Series: Monografias em Ciência da Computação, 9/89

ASPECTS OF NATURAL LANGUAGE QUERIES TO DATABASE REVISITED

Clarisse S. Souza
Eliane B. S. Carvalho

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 — CEP 22453
RIO DE JANEIRO — BRASIL

PUC/RJ - DEPARTAMENTO DE INFORMÁTICA

Series: Monografias em Ciência da Computação, 9/89

Editor: Paulo Augusto Silva Veloso

May, 1989

ASPECTS OF NATURAL LANGUAGE QUERIES TO DATABASE REVISITED

Clarisse S. Souza
Eliane B. S. Carvalho

This work has been partially sponsored by FINEP

In charge of publications:

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC RIO, Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22453 - Rio de Janeiro, RJ
BRASIL

Tel.:(021)529-9386 TELEX:31078 FAX:(021)274-4546
BITNET: userrtlc@lncc.bitnet

ASPECTS OF NATURAL-LANGUAGE QUERIES TO DATABASE REVISITED

de Souza, Clarisse S. & de Carvalho, Eliane B.S.

Departamento de Informatica
Pontificia Universidade Catolica do Rio de Janeiro
Rua Marques de Sao Vicente 225
22453 - Rio de Janeiro, RJ
Brazil
e-mail: usercsds@lncc.bitnet

ABSTRACT

Natural Language Interfaces for Database enquiry have lost most of their appeal since discourse theory entered the field of Artificial Intelligence and proved the need for solid pragmatic processing in intelligent computer-human interfaces. However, Database Systems are not likely to disappear in the near future and man-machine communications in this environment still calls for improvement. The present paper reports the results of research done along the line of syntax-oriented natural language processing in DB enquiries, emphasizing portability issues, and introduces Determination Grammars as a means to develop DB frontends.

1. INTRODUCTION

Natural Language Understanding Systems (NLUSs) for Database enquiry have apparently lost their position in the rank of theoretically-interesting computational linguistics research topics. A major field for investigation in early Artificial Intelligence projects involving natural language (NL), such systems are currently confined to a restricted sub-area of applied NL research. They seem to have been completely absorbed by a more comprehensive project: that of Intelligent Computer- Human Interfaces (ICHI) to Knowledge-Based Systems (KBSs).

An important change that may signal this loss of prestige is perhaps that of talking about Question-Answering Systems (QASs), as a whole, instead of pure Database Systems, in a more restrictive sense. QASs encompass both data-base and knowledge-base systems and once it comes to Intelligent Interfaces, a knowledge-based backend system is usually assumed.

Intelligent interaction in QAS environments presupposes the understanding of discourse situations and responding to these in an adequate manner. Essentially, communication is viewed in a behavioral framework, with psychosocial components. Such interaction must qualify in a number of requirements, so that one can say that there is a real DIALOGUE going on. Grice [1975] states such requirements in terms of cooperative principles; Allen & Perrault [1980] do it in terms of recognizing users intentions and plans. In fact, despite some differences in approach, intelligent communication is defined as that in which meaning is derived from the global speech situation, and not from sentences as individually uttered [Stucky,1987].

Natural Language is assumed as the prime means of intelligent communication, although not an exclusive one. We recognize that intelligent interactions do go on without speech, but we also accept that the role of language in communicative exchanges is very special. If we concentrate on computer-human intelligent interaction, this role is even more essential because man-machine exchanges must be language-based (though, of course, not natural language-based). The question to be asked is: what language system supports this interaction?

The expected performance of NLUSs in Database enquiries should, therefore, be pragmatically adequate, in the sense of offering adequate behavioral response to questions. Some consideration on the essence of Database Systems is, then, called for. We assume a Database System to be composed of a structure for storing pieces of information in a principled way and of a management agent that manipulates such structure in search of information. The retrieval process - and, for the sake of the present discussion, the update process too - is made according to constraints imposed by the DB structure and informational content. The system is typically viewed by the user as a passive deposit of data whose informational content is inferred by him once the data is retrieved. We emphasize the fact that cognitive computations are performed by the user, outside the scope of the system.

Knowledge-based Systems, on the other hand, are not a structured deposit of data and a management component, they are a set of processable facts and rules. This feature turns KBSs into active cognitive processors, able to derive inferences of a certain level. The result of such processing is more than a change of quality in the information retrieved: when some degree of reasoning is found in the system, it becomes an agent itself. Interaction between agents - be they artificial or natural - is quite different from interation between an agent and a non-agent. This, in the scope of the present paper, is the main distinctive feature of knowledge-based systems as compared to database systems.

The question now arises: is it possible to have a real DIALOGUE - i.e. an intelligent linguistic interaction - with Database Systems?

The answer can be elaborated along two different lines of reasoning. One is to believe it is possible - and, to that effect, desirable - provided that a frontend system is built to play the role of the artificial agent. The other is the opposite: i.e. to believe that DBSs should remain the same in essence and that computer-human interaction in this case should be measured and enhanced according to conventional man-machine communication criteria and goals.

Adopting a natural language-based dialogue framework for ICHI with Database systems implies that a "prolocutor" - or spokesman - is construed around the database management system (DBMS). This shell is provided with a series of models about the database, the user, the DBMS, the surrounding world and the dialogue structure itself. In particular, a powerful explanations module should be designed to make explicit the reasons why despite the fact that some questions can be asked and properly understood, they cannot possibly be answered because the backend system is - by definition - a passive deposit of information and not a reasoning agent. Cooperative behavior by the frontend becomes a crucial point, as extensively discussed by Kaplan [1984].

Adopting an artificial language-based framework for CHI with Database systems assumes that at one end of the communication line is a machine, or a non-agent according to the reasoning- ability criterion. A possible implication of such assumption could be that of discarding natural-language processing altogether, on the basis that query-languages satisfy the requirements of user-system exchanges. However, the reverse question could be asked: what sub-set of natural-language would apply to this case and why? An answer to this would probably involve research about what illocutionary acts - with some adaptation of Searle's [1975] original proposal, for example - are present in computer-human interactions.

The remainder of this paper presents the results of investigations along the latter approach, emphasizing syntactic aspects of the language used (or presumed to be used) in Database Question-Answering Systems.

2. DETERMINATION GRAMMARS

Experiments carried out with DBS users [de Souza,1988a] have shown that, among the problems encountered in the information retrieval process, knowing the syntax (and semantics) of the query language (QL) is an important one. First, users don't always know the correct syntax of the QL to formulate their queries and these are often rejected or turned into other queries whose syntax is known, instead. Second, because the semantics of the QL is even more deeply ignored than its syntax, inefficient queries - not to mention the really wrong ones - are common in this environment.

It must be remarked that DBS users are not typically programmers, which explains the point just made. For example, users might formulate queries like [a] and [b], below.

[a] Find FILE-#1 with Employee-name=Smith and Department-name=DP

Reject if project-name not equal P1 or P2

Display Employee-name Project-name

where the highlighted line shows a possible syntactic mistake in a hypothetical query language, in which the if-statement syntax should be: If project-name not equal P1 or P2 reject.

[b] Find FILE-#1 with Department-name=DP

If project-name not equal P1 or P2

Display Head-of-Dep

where the highlighted lines show a serious semantic mistake on the part of the user, if we consider that the head of the department remains the same, no matter what projects are involved in the query: the if-statement, though performed, has no influence whatsoever on the retrieved information. It might be argued that such queries are totally unlikely. A survey of actual DB queries in a real-world situation has shown, however, that this is not as rare as one might think [de Souza,1988,a]. The impact of such mistakes on the environment is clearly enormous, especially in large organizations.

For the great majority of DBS users, the retrieval activity is far more frequent than the update one - which can be non-existent. Therefore, a retrieval language that would handle problems such as [a] and [b] would benefit both users and DB administrators. Such a language should have the easiest syntax to learn and use and should keep users as far as possible from programming the query. It seems that both features converge to a sort of language in which the user expresses his information needs, and retrieval is performed in an optimized way. Would natural language qualify as such?

Asking this question is equivalent to asking another one: to what extent can natural language be used as an artificial code to communicate with machines? In other words: can a language such as Portuguese or English be deprived of psychosocial components and be reduced to syntax?

Determination Grammars [de Souza,1988a,b] are a special- purpose formalism to build NLUSs for Database enquiries, in which natural language [i.e. portuguese] is used as a code for man-machine communication. It is a "bottom-up" approach to enhancing CHI in conventional DB environments in search of a dual goal: in terms of DBSs, Determination Grammars (DGs) propose a (pseudo)natural means of communication with systems that is easy to use and efficient. In terms of computational linguistics, the approach reveals the boundaries of a syntax-oriented NL processing and proposes a principled methodology for the development of facilitated, though not fully intelligent man-machine interaction.

Determination Grammars aim at processing both grammatical and ungrammatical input. Grammatical input would be natural- language translations of queries of the type mentioned in [a] and [b], above. These queries could turn into questions such as [c] or [d], below.

[c] In which projects, besides P1 and P2, does Smith - of the Data Processing Department - take part?

[d] Who is the head of the DP Department?

Ungrammatical input could be of two different types: type 1 is that of questions in which there is a language mistake and type 2 is that of questions asked in a telegraphic style. Examples [e] and [f] show what these could look like.

[e] Which projects does Smith - of the Data Processing Department - *takes part in*, besides P1 and P2?

[f] Projects not P1 or P2 that Smith, from DP, works in.

Questions such as [f] are likely to be quite frequent as long as the input has to be typed at a keyboard. Grammatical questions may be extensive to type and users are expected to abbreviate typing in several ways.

The common substratum of grammatical and ungrammatical questions is the set of key words that refer to the database fields and values, together with those that refer to the retrieval operations desired by the user. Strict adhesion to this view can be found in the early naive DB natural-language interfaces that soon proved their limitations, back in the 1960s. However, the so-called head-and-modifier approach, that is based on the structural determination principle of natural language, as stated by European structuralist linguists such as Hjelmslev [1968], for example, provides a theoretical framework in which a combination of semantic keywords and syntactic phenomena can be used to specify a grammar for language understanding.

Determination Grammars assume a mapping between the Database entities, relationships and attributes and their linguistic counterparts (or "wording") is always possible. Moreover, DGs also assume that a mapping between retrieval operations on data have linguistic realizations as well, which means that two levels of vocabulary exist in a DG lexicon: a domain-dependent vocabulary (reflecting the DB content) and a domain-independent vocabulary (reflecting primary retrieval operations like *show, find, except, and, or,* and others). Besides these, there is still another class of words that should be included in the lexicon: words that are linguistically functional, such as pronouns, connectives and others, that signal syntactic operations such as anaphora, relativization, coordination, and so on. These, too, are domain-independent, by definition, since their semantic content is syntatically determined.

The DG formalism, thus, assigns semantically-motivated syntactic classes (or categories) to the various significant words of the application. For example, all the names of entities in the DB belong to a category N@, where @ is an index to the semantic type of the entity (an identifier) and N is the syntactic category equivalent to name-of-field, whose behavior can be predicted in terms of the determination principle. This principle applies recursively in a sentence, as shown below.

[g] Show all employees whose salary is greater than $1000.00.

Within the phrase **all employees whose salary is greater than $1000.00**, *whose salary is greater than $1000.00* modifies *employees*, and *all* modifies the rest of the phrase. Also, within the whole sentence, *all employees whose salary is greater than $1000.00* modifies *show*. A detailed explanation of this principle (the determination principle) is found in de Souza [1988,a].

In addition to semantically-motivated classes, there are also classes that are purely syntactically-motivated, as for example that of pronouns. So, the terminal symbols of a DG are the set of both classes. Words that are likely to appear in sentences, but not relevant by either of the criteria above, are discarded at input-reading. This is the case of phrasings such as *I would like to* or *Please, give me* and the like.

DG rules operate bottom-up and are of two kinds: meta-rules and rules proper. Meta-rules, as their name suggest, are control rules for the application of blocks of rules. For example, attribute-value determination rules apply before retrieval operation-scope determination rules. According to their nature, rules can be: context-free, context-sensitive or transformational. Context-free rules build complex constituents of the sentence structure; context-sensitive rules do not. This is a point in which DG terminology departs from standard formal language theory: DG context-sensitive rules are decision-rules for ambiguous terminal or non-terminal constituents. An example of the purpose and nature of such rules is found in [h'] and [h''] below. Contrast the two phrases:

[h']

... of the *two* employees ...

[h''] ... *two* of the employees ...

In [h'] the word two has an additional anaphoric value that derives from the context where it occurs, different from that which the same word has in [h'']. So, it should be the case that the *twos* belong to different categories. A DG context-sensitive rule for this would be stated as:[1]

[i] [context1.1] two [context1.2] -> [context1.1] Y [context1.2]

whereas another rule would state:

[j] [context2.1] two [context2.2] -> [context2.1] X [context2.2]

No aggregation of complex constituents is done via context-sensitive rules. This is not exactly a constraint of the formalism, but an idiosyncrasy of the applications in which DGs have been tested. [2]

Transformational rules are of two different types: declarative and procedural. Declarative transformational rules are those that, for example, perform deletions of symbols along the analysis. It is sometimes the case, for instance, that once a symbol is disambiguated it proves to be irrelevant in the current context. Deletion rules eliminate such symbols and guarantee expressive economy in the resulting structure. Another kind of transformational rule is that of copy: all anaphoric processes involve at least one such rule. However, a copy is essentially a procedural rule, in that the creation of a new constituent is performed in the middle of the parsing process.

3. DETERMINATION GRAMMARS PORTABILITY

If DGs are to support processing of a (pseudo)natural query language, the problem of portability becomes a crucial one. It is not reasonable that for every different domain the NLUS be totally rebuilt by some language-engineer. Portability is the main obstacle for the so-called semantic-grammars [Burton,1976], in that their domain-dependence demands a series of ad-hoc solutions to linguistic problems.

However, Determination Grammars present a methodology for the development of NLUS, in which mappings take place in a principled way. If principles are correctly formalized, then it is possible to suppose that the mapping of domain-dependent elements onto DG categories can be done automatically. If so, the result of such automation would be a proto-grammar, requiring only a few adjustments to accommodate occasional idiosyncrasies.

In fact, the DG-generating system should incorporate some kind of meta-grammar, reflecting general DG assumptions and functioning - i.e. an implementation of the underlying methodology. Additionally, special attention must be given to the assignment of categories to DB components. This assignment is the very basis of the whole process, since DGs are semantic grammars.

In order to test this hypothesis, a case study has been carried out [de Carvalho,1989]. A bibliographic reference DB prototype has been built and used to simulate a backend system with many complex relationships and attributes associated with each entity. An extended ERA model [Rego,1988] of the prototype DB provided the necessary input to the generator. Also, a "guiding" system has been assumed, in which the user was prompted to provide the appropriate lexical items corresponding to all the Data Model objects. Actually, this "guiding" system was in charge of carrying out the mapping procedure between the Data Model and the NLUS lexicon. Templates should be able to standardize the lexical entries in their optimal form.

A generator has been elaborated, incorporating a table of general syntactic categories, a meta-grammar (including the overall DG methodology) and a data-model processor, whose output is an image of the DB entities, relationships and attributes in a determination-like format.

[1] Note that, because of the assumed bottom-up parsing strategy, rules are stated in the recognition direction: i.e. the left-hand side is the input to rewriting and the right-hand side is the resulting output, once the rule is applied.

[2] Another use of context-sensitive rules is that of isolating a set of classes under a transient label, so that the remainder may be operated on without deteriorating the syntactic environment of the isolated classes, which may have to be preserved and restored, later on, to produce correct structuring of the input.

A DB-prototype has been used to simulate the backend system for a DG processor. An extended ERA model [Rego,1988] of the DB provided the necessary input to the generator. It was expected that the output of the generator would be a set of rules, meta- rules and a lexicon, that would properly analyse primary queries to the DB. Note that this schema puts a heavy load on the modelling activity: a poorly modelled DB would provide an inadequate grammar for the NLUS.

The case study has shown that a few assumptions have been wrong. In the first place, we expected the generated DG to be a specialization of the meta-DG incorporated in the generating system. But, contrary to this expectation, the gap between the two has proved to be very short. In fact, specialization was typically derived from lexical idiosyncrasies and not from syntactic ones, which means that the rules themselves could apply to many different interfaces. Second, because of this, the generation problem had to be stated in different terms: the specialization hypothesis was not the correct one, but rather that of generalization - not of an initial set of rules, but of the original statement of Determination Grammars themselves.

As can be seen in the original proposal [de Souza,1988,a,b], Determination Grammars were conceived as a methodology that would derive different sets of rules, for different applications as one migrates from one to the other. The difference in the rules would be caused by domain-dependent relationships between components of the DBs, which means that initially domain-dependence would affect syntax itself. The results of the experiment have shown that domain-dependence can be confined to the lexicon, preserving syntactic rules from major changes. Once the mapping between DB components and lexical-entries is performed, the grammar can be used in various domains. This mapping, on its turn, is automatically done according to the determination principle that governs DGs. Thus, automation is possible.

The consequences of such automation both on the quality of the NLUS and on the quality of the necessary data- model call for further investigation. Another important topic is that of adequately mapping queries onto various DB files, in a complex environment where information is to be found in different physical archives. However, this is a mainly DB-structure problem and not a basically linguistic one [Wallace,1985].

4. DISCUSSION OF THE CASE STUDY RESULTS

Although the experiment does not provide enough testing of the hypotheses underlying Determination Grammars, results point out relevant research topics. Among these, we will concentrate our discussion on: (1) what is the type of language that DGs specify and why, and (2) what is the linguistic coverage of the NLUSs derived.

The type of language specified by Determination Grammars is constrained by the fact that the domain must be modelled by an ERA-like methodology. This can be illustrated by the fact that verbs, for example, are typically transitive, in that they express relationships between entities of the model. Moreover, the automatic class-assignment procedure of the generator is only prepared to process this type of model . Therefore, the answer to the question asked above is that DGs specify languages that are used to express information retrieval needs from a domain that is modelled via ERA or a similar methodology. It comes down to saying that such languages are typically database-query languages.

As to the second question, the linguistic coverage achieved by the experiment includes adequate processing of ellipsis, metonymy, coordination and relativization. The importance of such phenomena in natural language use is clear and some remarks may have to be added about metonymy. DGs have been used only for the Portuguese language, in which metonymy is a very frequent feature. We know this is not true of other languages, but - nevertheless - it seems important that the formalism accomodates this kind of structure.

Previous DG-based NLUSs, such as mentioned in de Souza [1988,a], incorporate more processing capacity than that achieved in the experiment. In those, besides the above mentioned syntactic features, negation, anaphora (simple and complex) and special structures such as the expression of numeric intervals and computations have been adequately analysed. In fact, since the grammar itself was built manually, the inclusion of special cases was easier to accommodate in the overall framework. However, it must be said that the case study had the objective of testing general automation hypotheses. Because of this, many phenomena of natural language interactions with DBs have not been extensively tested, which does not mean they are not possibly handled. This is the case of anaphora, of the so-called "telegraphic questions" and of a few other structures.

The most important point to be made in the present discussion is that of the derivation of the lexicon from the DB model. We have claimed that this process can be automated, and in our case study it has actually been. However, some adaptation of a pure ERA-like model has to be assumed. This is due to facts such as the following:

- the model may contain something like empl-add, which stands for employee-address. If this equivalence is not properly handled, all questions involving this attribute become unprocessable, since it is unreasonable to expect the users to know what kind of abbreviation was made in the physical DB..

- relationships in the model are not always expressed by specific labels. This amounts to assigning words - often verbs - to express them in NL questions.

- verbs that bind an entity to its attributes, like employee "works at" department-x, are not present in the model, either.

- some other implicit expressions in the model are, for example, those that refer to units associated to quantified data. This is the case of dollar in salary, years in age, and so on.

- additionally, the model does not indicate which, among all attributes of an entity, are possibly present in a metonymy. For example: we may refer to the americans, and not necessarily to the american employees. In the former, the entity employee has been ommitted and the attribute nationality encompasses the whole meaning of the expression in a typical part-whole relationship.

The way in which the working prototype we have developed solved the problems stated above is that of using the modelling methodology as a guide to prompt the user so that he or she provides the necessary information. For example: for each item of the data-model, the user is asked to state the complete name and the synonyms or abbreviations he or she desires to include.

In terms of verbs or implicit expressions in the model, the user makes them explicit as the interface formulates the adequate questions to obtain such words. For example: there is a relationship between employee and department; the interface might prompt the user as : An employee _______ department. (fill in with the adequate expression). This would provide the verb and optional prepositions or particles. The same procedure would apply to the numeric fields and their units.

The treatment of metonymy is more complex. In fact the interface in the prototype hypothesized that all attributes that could fill the template <ENTITY> is <ATTRIBUTE> were potentially usable in a metonymic structure. In fact, the verb in the template could be any copula.

It may be argued that this customization process is extremely extensive and time-consuming. However, it must be remarked that this activity needs to be done once, at the setting of the NLUS, and not anymore, until another data-model updating. Database updates could be accommodated in localized user- prompting procedures about the insertion or alteration of data.

5. CONCLUSION

The reported investigation on the use of natural language for database enquiry should be considered within the limits of the theoretical assumptions stated in section 1. In other words, there is no intention of building an intelligent processor to mediate the communication between user and system in a DB environment. This is not due to any disagreement with the fact that such processors can be built, or that they should be built, but rather to the fact that there may be reasons for preserving the original structure and purpose of conventional databases. Among such reasons, we may select that of the difficulty of both creation and maintenance of large knowledge bases. It does not mean that this problem is unsolvable, but that the needs provided for by conventional DBs are likely to remain satisfied by DBs for some period of time in the future.

Within this context, the problem of natural language frontend portability and DB domain-dependence is a major one. So, all solutions that satisfy the requirement of NL question- answering with DBs must envisage portability factors. The partial automation of grammars - i.e. the customization process - is then a desirable feature of all proposed formalisms.

The results presented in this paper have assumed that natural language can be used to some extent as an artificial code for man-machine communications, provided that there is a clear statement of the principles that apply to the situation. Determination Grammars, originally a methodological proposal, have proved to be formalizable so as to allow for the implementation of a general syntactic processor. This processor adequately analyses natural-language queries to Databases.

It may be argued that the bottom-up approach that DGs assume, going from the idea of artificial query languages with rigid syntax to that of flexible (pseudo)natural query languages, is too restrictive. However, two points must be made in this respect. One is that, although leaving aside a genuine discourse grammar and a variety of models that would account for fully cooperative behavior, the DG-based query language is far more user-friendly than any other conventional artificial query language. Besides, it also provides an interesting opportunity for query optimization, so that the DB environment is globally affected by the use of such code (in which programming is kept outside the user's reach).

Another point is that this kind of pseudo-natural.interaction reveals the boundaries of syntax in language use. Note that this approach privileges the syntactic power of NL, very much like some theoretical linguistic proposals have done back in the 1960s (namely, those of early generative theory). In other words: it assumes that all - or most of - the semantics of utterances is derivable from syntactic structure and lexical information alone. In linguistic theory, this approach has been extended and revised (or even abandonned by some) because human language use presents clear evidence of the importance of context and extra-grammatical information in meaningful communication. However, systems are not human agents, and one cannot expect their behavior to be the same as ours. Therefore, it seems reasonable to think that linguistic theory as such would not necessarily apply to computer-human interactions. Some characteristics of natural language - for example, some sorts of illocutionary acts - are not likely to be found in this environment. And, if this is true, a bottom-up approach to natural language processing could point out some interesting features of this unique dialogue situation.

We believe that Determination Grammars offer an interesting topic for computational linguistics research and for database interfaces, as well. The resulting NLUS may be extended for a variety of DB applications, including that of evaluating the data-model itself. For instance, if a certain question is not correctly processed by the grammar, it may be the case that the data-model is inadequate. We intend to explore DGs along its main line of assumptions and eventually extend them to accommodate some discourse phenomena that are clearly signalled by syntax. This should enhance cooperation in DG-based frontends and provide adequate user-friendly interfaces to Database enquiry.

6. REFERENCES

Allen,J.F. & Perrault,C.R. (1980) Analyzing Intentions in Utterances. Artificial Intelligence 15, 3.

Burton,R. (1976) Semantic Grammar: An Engineering Technique for Constructing Natural Language Understanding Systems. BBN Technical Report no.3453.

de Carvalho,E.B.S. (1989) Estudo sobre a Automatizacao do Processo de Desenvolvimento de uma Gramatica de Determinacao Basica para Acesso a Bancos de Dados. Unpublished Monograph. PUC/RJ, Depto. de Informatica.

de Souza,C.S. (1988,a) Gramaticas de Determinacao: uma Ferramenta para o Processamento da Lingua Portuguesa. Ph.D. Dissertation. PUC/RJ, Depto. de Letras.

de Souza,C.S. (1988,b) Gramaticas de Determinacao: uma Proposta Metodologica. Proc. of the Jornadas Argentinas de Informatica e Investigacion Operativa, vol.2.

Grice,H.P. (1975) Logic and Conversation, in P.Cole and J.L. Morgan (eds) Syntax and Semantics:Speech Acts, Vol.3, New York, Academic Press.

Hjelmslev,L. (1968) Prolegomenos a uma Teoria da Linguagem. Brazilian Edition,1975) Sao Paulo, Editora Perspectiva.

Kaplan,S.J (1984) Designing a Portable Natural Language Database Query System. ACM Transactions on Database Systems, vol.9, no.1.

Rego,S.P. (1988) Modelagem Conceitual de Dados: uma Comparacao entre Abordagens. Proc. of Congresso Nacional da SUCESU, SUCESU, Rio de Janeiro.

Searle,J.R. (1975) Indirect Speech Acts, in P.Cole and J.L. Morgan (eds) Syntax and Semantics:Speech Acts, Vol.3, New York, Academic Press.

Stucky,S. (1987) The Situated Processing of Situated Language. CSLI Report no.CSLI-87-80, Stanford University.

Wallace,M. (1985) Communicating with Databases in Natural Language. New York, E. Horwood.