International Journal of Computer Science and Network (IJCSN)
Volume 1, Issue 4, August 2012 www.ijcsn.org ISSN 2277-5420

A Study of the Various Architectures for Natural Language Interface to DBs

[1]B.Sujatha, [2]Dr.S.Viswanadha Raju, [3]Humera Shaziya

[1]Research Scholar, Dept. of CSE
JNTUH, Hyderabad, AP, India

[2]Professor & Head, Dept. of CSE
J.N.T.University, Jagtial

[3]Lecturer in Computers, Dept. of M.C.A
Nizam College, Hyderabad

Abstract

This paper is an introduction to the architecture of the natural language interfaces to databases (NLIDBS). First the concept of Intelligent Databases (IDBS) is presented. Some advantages and disadvantages of NLIDBS are then discussed followed by the discussion of the components of NLIDB. Comparison of NLIDBS to formal query languages, form-based interfaces, and graphical interfaces are then discussed. The discussion then moves on to NLIDB architectures in which various architectures are discussed.

Keywords: *IDBS, Linguistics Component, Symbolic Approach, Empirical Approach, Pattern Matching System, Syntax Based System, Semantic Grammer System.*

I. INTRODUCTION

A natural language interface to a database (Nlidb) is a system that allows the user to access information stored in a database by typing requests expressed in some natural language (e.g. English and Telugu). The purpose of this paper is to serve as an introduction to some key concepts, problems, methodologies, and lines of research in the area of natural language interfaces to databases. This paper is by no means a complete discussion of all the issues that are relevant to NLIDBS. Although the paper contains hints about the capabilities of existing NLIDBS, it does not contain complete descriptions of particular systems, nor is the purpose of this paper to compare Particular NLIDBS. This paper is mainly based on information obtained from published documents. The authors do not have personal hands-on experience with most of the NLIDBS that will be mentioned. Whenever a system's feature is mentioned, this means that the documents cited state that the particular system provides this feature, and it is not implied that other systems do not have similar capabilities. Finally, this paper assumes that the user's requests are communicated to the Nlidb by typing on a computer keyboard. Issues related to speech processing are not discussed. The remainder of this paper is organized as follows: In section 2 a brief overview of the intelligent database system (IDBS) is discussed. Section 3 talk about the components of NLIDB; Section 4 contains discursion on the advantages and disadvantages of NLIDBS; Section 5 presents various approaches to interface to database; Section 6 presents some of the architectures of NLIDBS. The paper ends with a Conclusion.

II. INTELLIGENT DATABASE SYSTEM (IDBS)

An IDBS is endowed with a data management system able to manage large quantities of persistent data to which various forms of reasoning can be applied to infer additional data and information. This includes knowledge representation techniques, inference techniques, and intelligent user interfaces – interfaces which extend beyond the traditional query language approach by making use of natural language facilities. These techniques play important role in enhancing databases

systems : knowledge representation techniques allow one to represent better in the DB the semantics of the application domains, inference techniques allow one to reason about data to extract additional data and information, Intelligent user interfaces help users to make requests and receive the replies. Intelligent databases systems are the systems that manage information in a natural way, making that information easy to store, access and use. One of the main reasons for using intelligent database system is that we live in a state of Information glut. To simply survive in today's society, we need to access and use this information. By using intelligent databases system we can have better access to, and use of, more kinds of information that they could otherwise. This means intelligent databases systems should provide high-level intelligent tools that provide new insights into the contents of the database by extracting knowledge from data. Make information available to larger numbers of people because more people can now utilize the system due to its ease of use. Improve the decision making process involved in using information after it has been retrieved by using Higher level information models Interrelate information from different sources using different media so that the information is more easily Absorbed and utilized by the user. Use of knowledge and inference, making it easier to retrieve, view and make decisions with information. In recent times, there is a rising demands for non-expert users to query relational databases in a more natural language encompassing linguistic variables and terms, instead of operating on the values of the attributes. Intelligent interface for database systems, a promising approach, enhance the users in performing flexible querying in databases. The research and advancement of NLIDB, an important step towards the development of intelligent databases system and it has emerged as a new discipline and have fascinated the attention to number of researchers. The first work on natural language interfaces (NLIs) was done by Warren Weaver in 1947 with translation systems. At the end of the 70s, Victor Yngve of MIT proposed a grammatical method for NLP based on dictionaries. In the early 70s, in Cambridge, Leningrad, Grenoble, and Texas some work were done on the "interlingua" approach: the idea that any natural language can be expressed in a universal representation. Heavily criticized, this idea, impossible to validate, was the origin of "knowledge representation." It also helped to conclude that NLP needed more knowledge than pure syntax of the language. After that, a new era of semantic processing (based on semantic rather than syntactic patterns) was pioneered by Wilks, Weinzenbaum (Eliza and Doctor developed in 1966), and Colby (Parry implemented in 1975). Another branch of this idea tried to associate formal systems with NLP; examples are Student of Brobow (1968) and Baseball written by Chomsky, Green, Wolf, and Laughery. This system was one of the first database access systems. Other interesting projects are the following: SHRDLU by Terry Winograd (1972) suggested a procedural representation of sentences; Margiede Roger Schank (around 1970) used conceptual dependences to represent sentences. Natural Language Interfaces is a hot area of research since long. The purpose of Natural language Interface to Database System is to accept requests in English or any other natural language and attempts to 'understand' them or we can say that Natural language interfaces to databases (NLIDB) are systems that translate a natural language sentence into a database query. Although the earliest research has started since the late sixties, NLIDB remains as an open research problem. A complete NLIDB system will benefit us in many ways. Anyone can gather information from the database by using such systems. Additionally, it may change our perception about the information in a database. Traditionally, people are used to working with a form; their expectations depend heavily on the capabilities of the form. NLIDB makes the entire approach more flexible, therefore will maximize the use of a database. There are many applications that can take advantages of NLIDB. In PDA and cell phone environments, the display screen is not as wide as a computer or a laptop. Filling a
form that has many fields can be tedious: one may have to navigate through the screen, to scroll, to look up the scroll box values, etc. Instead, with NLIDB, the only work that needs to be done is to type the question similar to the SMS (Short Messaging System).

III. COMPONENTS OF NLIDB

Computing scientists have divided the problem of natural language access to a database into two sub-components

A. Linguistic Component

It is responsible for translating natural language input into a formal query and generating a natural language response based on the results from the database search.

B. Database Component

It performs traditional Database Management functions. A lexicon is a table that is used to map the words of the natural input onto the formal objects (relation names, attribute names, etc.) of the database. Both parser and semantic interpreter make use of the lexicon. A natural language generator takes the formal response as its input, and inspects the parse tree in order to generate adequate natural language response. Natural language database systems make use of syntactic knowledge and knowledge about the actual database in order to properly relate natural language input to the structure and contents of that database. Syntactic knowledge usually resides in the linguistic component of the system, in particular in the syntax analyzer whereas knowledge about the actual database resides to some extent in the semantic data model used. Questions entered in natural language translated into a statement in a formal query language. Once the statement unambiguously formed, the query is processed by the database management system in order to produce the required data. These data then passed back to the natural language component where generation routines produce a surface language version of the response.

IV. ADVANTAGES AND DISADVANTAGES

This section discusses some of the advantages and disadvantages of NLIDBS, comparing them to formal query languages, form-based interfaces, and graphical interfaces. Access to the information stored in a database has traditionally been achieved using formal query languages, such as SQL.

A. Advantages of NLIDBS

No artificial language: One advantage of NLIDBS is supposed to be that the user is not required to learn an artificial communication language. Formal query languages are difficult to learn and master, at least by non-computer-specialists. Graphical interfaces and form-based interfaces are easier to use by occasional users; still, invoking forms, linking frames, selecting restrictions from menus, etc. constitute artificial communication languages, that have to be learned and mastered by the end-user. In contrast, an ideal Nlidb would allow queries to be formulated in the user's native language. This means that an ideal Nlidb would be more suitable for occasional users, since there would be no need for the user to spend time learning the system's communication language. In practice, current NLIDBS can only understand limited subsets of natural language. Therefore, some training is still needed to teach the end-user what kinds of questions the Nlidb can or cannot understand. In some cases, it may be more difficult to understand what

sort of questions an Nlidb can or cannot understand, than to learn how to use a formal query language, a form-based interface, or a graphical interface (see disadvantages below). One may also argue that a subset of natural language is no longer a natural language.

Better for some questions: It has been argued (e.g. [28]) that there are kinds of questions (e.g. questions involving negation, or quantification) that can be easily expressed in natural language, but that seem difficult (or at least tedious) to express using graphical or form-based interfaces. For example, "Which department has no programmers?" (negation), or "Which company supplies every department?" (universal quantification), can be easily expressed in natural language, but they would be difficult to express in most graphical or form-based interfaces. Questions like the above can, of course, be expressed in database query languages like Sql, but complex database query language expressions may have to be written.

B. Disadvantages of NLIDBS

Linguistic coverage not obvious: A frequent complaint against NLIDBS is that the system's linguistic capabilities are not obvious to the user. As already mentioned, current NLIDBS can only cope with limited subsets of natural language. Users find it difficult to understand (and remember) what kinds of questions the NLIDB can or cannot cope with. For example, Masque is able to understand "What are the capitals of the countries bordering the Baltic and bordering Sweden?", which leads the user to assume that the system can handle all kinds of conjunctions (false positive expectation).

However, the question "What are the capitals of the countries bordering the Baltic and Sweden?" cannot be handled. Similarly,
a failure to answer a particular query can lead the user to assume that "equally difficult" queries cannot be answered, while in fact they can be answered (false negative expectation). Formal query languages, form-based interfaces, and graphical interfaces typically do not suffer from these problems. In the case of formal query languages, the syntax of the query language is usually well-documented, and any syntactically correct query is guaranteed to be given an answer. In the case of form-based and graphical interfaces, the user can usually understand what sorts of questions can be input, by browsing the options offered on the screen; and any query that can be input is guaranteed to be given an answer.

V. VARIOUS APPROACHES

Natural language is the topic of interest from computational viewpoint due to the implicit ambiguity that language possesses. Several researchers applied different techniques to deal with language. Next few sub-sections describe diverse strategies that are used to process language for various purposes.

A. Symbolic Approach (Rule Based Approach)

Natural Language Processing appears to be a strongly symbolic activity. Words are symbols that stand for objects and concepts in real worlds, and they are put together into sentences that obey well specified grammar rules. Hence for several decades Natural Language Processing research has been dominated by the symbolic approach (Miikkulainen, 1997).R. Akerkar and M. Joshi Knowledge about language is explicitly encoded in rules or other forms of representation. Language is analyzed at various levels to obtain information. On this obtained information certain rules are applied to achieve linguistic functionality. As Human Language capabilities include rule-base reasoning, it is supported well by symbolic processing. In symbolic processing rules are formed for every level of linguistic analysis. It tries to capture the meaning of the language based on these rules.

B. Empirical Approach (Corpus Based Approach)

Empirical approaches are based on statistical analysis as well as other data driven analysis, of raw data which is in the form of text corpora. A corpus is collections of machine readable text. The approach has been around since NLP began in the early 1950s. Only in the last 10 years or so empirical NLP has emerged as a major alternative to rationalist rule-based Natural Language Processing. Corpora are primarily used as a source of information about language and a number of techniques have emerged to enable the analysis of corpus data. Syntactic analysis can be achieved on the basis of statistical probabilities estimated from a training corpus. Lexical ambiguities can be resolved by considering the likelihood of one or another interpretation on the basis of context. Recent research in computational linguistics indicates that empirical or corpus –based methods are currently the most promising approach to developing robust, efficient natural language processing (NLP) systems (Church, 1993; Charniak, 1993). These methods automate the acquisition of much of the complex knowledge required for NLP by training on suitably annotated natural language corpora, e.g. tree-banks of parsed sentences (Marcus, 1993). Most of the empirical NLP methods employ statistical techniques such as n-gram models, hidden Markov models (HMMs), and probabilistic context free grammars (PCFGs). Given the successes of empirical NLP methods, researchers have recently begun to apply learning methods to the construction of information extraction systems (McCarthy, 1995), (Soderland, 1995), (Riloff, 1993, 1996), (Huffman, 1996). Several different symbolic and statistical methods have been employed, but most of them are used to generate one part of a larger information extraction system. (Majumder, 2002) experimented N-gram based language modeling and claimed to develop language independent approch to IR and Natur al Language Processing. 2.3 Connectionist Approach (Using Neural Network) Since human language capabilities are based on neural network in the brain, Artificial Neural Networks (also called as connectionist network) provides on essential starting point for modeling language processing (Wermter, 1997). In the recent years, the field of connectionist processing has seen a remarkable development.

VI. ARCHITECTURES

6.1. Pattern-matching systems

Some of the early NLIDBS relied on pattern-matching techniques to answer the user's questions. The main advantage of the pattern-matching approach is its simplicity: no elaborate parsing and interpretation modules (see later sections) are needed, and the systems are easy to implement. Also, pattern-matching systems often manage to come up with some reasonable answer, even if the input is out of the range of sentences the patterns were designed to handle. Returning to the example above, the second rule would allow the system to answer the question "Is it true that the capital of each country is Athens?", by listing the capital of each country, which can be considered as an indirect negative answer. Pattern-matching systems are not necessarily based on such simplistic techniques as the ones discussed above. Savvy, a pattern matching system discussed in [63] (p.153), employs pattern-matching techniques similar to the ones used in signal processing. According to [63], some pattern-matching systems were able to perform impressively well in certain applications. However, the shallowness of the pattern-matching approach would often lead to bad failures. In one case (mentioned in [63]), when a pattern-matching Nlidb was asked "titles of employees in los angeles.", the system reported the state where each employee worked, because it took "in" to denote the post code of Indiana, and assumed that the question was about employees and states.

6.2 Syntax-based systems

In syntax-based systems the user's question is parsed (i.e. analysed syntactically), and the resulting parse tree is directly mapped to an expression in some database query language. A typical example of this approach is Lunar Syntax-based NLIDBS usually interface to application-specific database systems, that provide database query languages carefully designed to facilitate the mapping from the parse tree to the database query. It is usually difficult to devise mapping rules that will transform directly the parse tree into some expression in a real-life database query language.

6.3 Semantic grammar systems

In semantic grammar systems, the question-answering is still done by parsing the input and mapping the parse tree to a database query. The difference, in this case, is that the grammar's categories (i.e. the non-leaf nodes that will appear in the parse tree) do not necessarily correspond to syntactic concepts. Semantic grammars were introduced as an engineering methodology, which allows semantic knowledge to be easily included in the system. However, since semantic grammars contain hard-wired knowledge about a specific knowledge domain, systems based on this approach are very difficult to port to other knowledge domains a new semantic grammar has to be written whenever the NLIDB is configured for a new knowledge domain.

6.4 Intermediate representation languages

Most current NLIDBS first transform the natural language question into an intermediate logical query, expressed in some internal meaning representation language. The intermediate logical query expresses the meaning of the user's question in terms of high level world concepts, which are independent of the database structure. In the intermediate representation language approach, the system can be divided into two parts. One part starts from a sentence up to the generation of a logical query. The other part starts from a logical query until the generation of a database query. In the part one, The use of logic query languages makes it possible to add reasoning capabilities to the system by embedding the reasoning part inside a logic statement. In addition, because the logic query languages is independent from the database, it can be ported to different database query languages as well as to other domains, such as expert systems and operating systems.

VII. CONCLUSION

Research is done from the last few decades on Natural Language Interfaces. With the advancement in hardware processing power, many NLIDBS mentioned in historical background got promising results. Though several

NLIDB systems have also been developed so far for commercial use but the use of NLIDB systems is not wide-spread and it is not a standard option for interfacing to a database. This lack of acceptance is mainly due to the large number of deficiencies in the NLIDB system in order to understand a natural language.

References

[1] J. Allen. Recognizing Intentions from Natural Language Utterances. In M. Brady and R.C. Berwick, editors, Computational Models of Discourse, chapter 2, pages 107–166. MIT Press, Cambridge, Massachusetts, 1983.
[2] H. Alshawi. The Core Language Engine. MIT Press, Cambridge, Massachusetts, 1992.
[3] H. Alshawi, D. Carter, R. Crouch, S. Pulman, M. Rayner, and A. Smith. CLARE – A Contextual Reasoning and Cooperative Response Framework for the Core Language Engine. Final report, SRI International, December 1992.
[4] I. Androutsopoulos. Interfacing a Natural Language Front-End to a Relational Database (MSc thesis). Technical paper 11, Department of Artificial Intelligence, University of Edinburgh, 1993.
[5] I. Androutsopoulos, G. Ritchie, and P. Thanisch. An Efficient and Portable Natural Language Query Interface for Relational Databases. In P.W. Chung, G. Lovegrove, and M. Ali, editors, Proceedings of the 6th International Conference on Industrial & Engineering Applications of Artificial Intelligence and Expert Systems, Edinburgh, U.K., pages 327–330. Gordon and Breach Publishers Inc., Langhorne, PA, U.S.A., June 1993. ISBN 2–88124–604–4.
[6] P. Auxerre. MASQUE Modular Answering System for Queries in English - Programmer's Manual. Technical Report AIAI/SR/11, Artificial Intelligence Applications Institute, University of Edinburgh, March 1986.
[7] P. Auxerre and R. Inder. MASQUE Modular Answering System for Queries in English - User's Manual. Technical Report AIAI/SR/10, Artificial Intelligence Applications Institute, University of Edinburgh, June 1986.
[8] B. Ballard and D. Stumberger. Semantic Acquisition in TELI. In Proceedings of the 24th Annual Meeting of ACL, New York, pages 20–29, 1986.
[9] B.W. Ballard. The Syntax and Semantics of User-Defined Modifiers in a Transportable Natural Language Processor. In Proceedings of the 22nd Annual Meeting of ACL, Stanford, California, pages 52–56, 1984.
[10] B.W. Ballard, J.C. Lusth, and N.L. Tinkham. LDC-1: A Transportable, Knowledgebased Natural Language Processor for Office Environments. ACM Transactions on Office Information Systems, 2(1):1–25, January 1984.
[11] M. Bates, M.G. Moser, and D. Stallard. The IRUS transportable natural language database interface. In L. Kerschberg, editor, Expert Database Systems, pages 617–630. Benjamin/Cummings, Menlo Park, CA., 1986.
[12] BBN Systems and Technologies. BBN Parlance Interface Software – System Overview, 1989.
[13] J.E. Bell and L.A. Rowe. An Exploratory Study of Ad Hoc Query Languages to Databases. In Proceedings of the 8th International Conference on Data Engineering, Tempe, Arizona, pages 606–613. IEEE Computer Society Press, February 1992.
[14] BIM Information Technology. Loqui: An Open Natural Query System – General Description, 1991. (Commercial leaflet).
[15] J.-L. Binot, L. Debille, D. Sedlock, and B. Vandecapelle. Natural Language Interfaces: A New Philosophy. SunExpert Magazine, pages 67–73, January 1991.
[16] R.J. Bobrow. The RUS System. In Research in Natural Language Understanding, BBN Report 3878. Bolt Beranek and Newman Inc., Cambridge, Massachusetts, 1978.
[17] R.J. Bobrow, P. Resnik, and R.M. Weischedel. Multiple Underlying Systems: Translating User Requests into Programs to Produce Answers. In Proceedings of the 28th Annual Meeting of ACL, Pittsburgh, Pennsylvania, pages 227–234, 1990.
[18] R.A. Capindale and R.G. Crawford. Using a Natural Language Interface with Casual Users. International Journal of Man-Machine Studies, 32:341–361, 1990.
[19] J.G. Carbonell. Discourse Pragmatics and Ellipsis Resolution in Task-Oriented Natural Language Interfaces. In Proceedings of the 21st Annual Meeting of ACL, Cambridge, Massachusetts, pages 164–168, 1983.
[20] S. Ceri, G. Gottlob, and L. Tanca. Logic Programming and Databases. Springer-Verlag, Berlin, 1990.
[21] S. Ceri, G. Gottlob, and G. Wiederhold. Efficient Database Access from Prolog. IEEE Transactions on Software Engineering, 15(2):153–163, February 1989.
[22] S. Ceri and G. Pelagatti. Distributed Databases: Principles and Systems. McGraw-Hill, New York, 1984.
[23] J. Clifford. Natural Language Querying of Historical Databases. Computational Linguistics, 14(4):10–34, December 1988.
[24] J. Clifford. Formal Semantics and Pragmatics for Natural Language Querying. Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, Cambridge, England, 1990.
[25] J. Clifford and D.S. Warren. Formal Semantics for Time in Databases. ACM Transactions on Database Systems, 8(2):215–254, June 1983.

[26] E.F. Codd. A Relational Model for Large Shared Data Banks. Communications of the ACM, 13(6):377–387, 1970.
[27] E.F. Codd. Seven Steps to RENDEZVOUS with the Casual User. In J. Kimbie and K. Koffeman, editors, Data Base Management. North-Holland Publishers, 1974.
[28] P.R. Cohen. The Role of Natural Language in a Multimodal Interface. Technical Note 514, Computer Dialogue Laboratory, SRI International, 1991.
[29] A. Copestake and K. Sparck Jones. Natural Language Interfaces to Databases. The Knowledge Engineering Review, 5(4):225–249, 1990.
[30] F. Damerau. Operating statistics for the transformational question answering system. American Journal of Computational Linguistics, 7:30–42, 1981.