Aneesah: A Conversational Natural Language Interface to Databases

K. Shabaz, Jim D. O'Shea, *Member, IEEE*,
Keeley A. Crockett, *Senior Member, IEEE,* A. Latham, *Member, IEEE*

Abstract **– This paper presents the design and development of a novel Natural Language Interface to Database (NLIDB). The developed prototype is called Aneesah the NLIDB, which is capable of allowing users to interactively/conversely access desired information stored in a relational database. This paper introduces the novel conversational agent enabled architecture of Aneesah NLIDB and describes the scripting techniques that has been adopted for its development. The proposed framework for Aneesah NLIDB is based on pattern matching techniques implemented to converse with users, handle complexities and ambiguities for building dynamic SQL queries from multiple dialogues in order to extract database information. The preliminary evaluation results gathered following a pilot study reveal promising results.**

Index Terms **– Natural Language Interface to Databases (NLIDB), Conversational Agents (CA), Knowledge base, Artificial Intelligence (AI), Pattern Matching (PM).**

I. Introduction

Much of largest sources of information storages in public or commercial environment are databases. The information is a key element required when making decisions. Retrieving information from a database normally requires querying the database using a specialised programming code called SQL query, which is not accessible to the ordinary users. In order to avoid the difficulty of using SQL language for inexperienced users, there exists a need for creating applications that permit users to access desired information stored in a database. Natural Language Interfaces to Databases (NLIDBs) are computer programmes, which replace the requirement to react with a skilled programmer to query a database by using natural conversation with an agent.

Recently, the demand for Natural Language Interfaces to Databases has risen in result of many users accessing information from different types of systems such as computers, tablets, and smartphones [1]. NLIDB development attempts cover a time span of over half a century, however there still remain unsolved key challenges for their wider acceptance in public and commercial environment. Despite many development approaches, there are number of challenges that have been identified by researchers such as linguistic problems, conversational abilities, query translation, result refinement, domain independence, and ease of configuration [2].

This paper is organized as follows: Section II & III will give background of NLIDB and Conversational Agent applications. Section IV will describe existing problems and challenges in developing NLIBDs followed by Section V explaining the knowledge engineering of domain. Section VI describes the structure of framework adopted for prototype NLIDB (Aneesah). Section VII describes the Pilot Study, results and discussion. Section VIII will conclude and highlight areas for further work.

II. Natural Language Interface to Database

The idea of a computer taking the role of a human in conversation was first proposed by Alan Turing [3] and later was put into practice in the 70s with the development of the ELIZA Chatbot, which simulated a psychotherapist [4]. Among early development of NLIDBs, the most popular NLIDB was LUNAR [5]. LUNAR was built to perform moon rocks analysis based on an underlying database but it had functional limitations and could not be generalized to other domains. Soon after, the development of RENDEZVOUS was intended to simulate open dialogues to users in order to devise database queries [6]. The users anticipated questions had to closely match for the system to understand entered text, for query processing. LADDER development targeted big data and distributed databases. It was expected that LADDER would require substantial customization (.e.g. new grammar, domain knowledge), for adopting new domains [9]. PLANES designed features were based on the principals of RENDEZVOUS, which used a flight and airport database to answer users' questions [10]. PLANES system also required a new grammar and extensive customization to be able to work with a different domain [9].

In 80s, several other systems were developed such as CHAT80 which translated natural language in Prolog language [10] and TEAM which translated natural language queries into Simple Object Database Access query language [11]. Other systems such as PARLANCE built to resolve domain configuration issues and allowed users to manually configure underlying domain [12]. ASK also appeared as a cross application NLIDB [13] that allowed users to input their requests in natural language to generate appropriate responses from underlying database [14]. The development on NLIDBs continued to evolve in 90s with research on various commercial systems such as CHILL [2], INTELLECT [15], DATALINKER, Q&A Symantec and LOQUI [16]. In recent years some online systems appeared such as Wolfram Alpha, Powerset and TrueKnowledge that

Manuscript received March 23, 2015.
All authors are with the Intelligent Systems Group,
Department of Computing & Mathematics, Manchester Metropolitan University, Manchester, M1 5GD, UK. Email:
{k.shabaz, j.d.oshea, k.crockett, a.latham}@mmu.ac.uk

were designed to rely and work on initial information imported in the development [17]. More recent developments have contributed less efforts to create interaction between user and NLIDB systems. ITG is another developed NLIDB that urged users to repeatedly validate predicted text as user enters his/her input [18]. C-Phrase was developed on Codd's tuple calculus to allow context free grammar [19] and NaLIR a keyword based search interface developed using Natural Language Processing [20].

III. Conversational Agents

Conversational Agents (CAs) are also computer dialogue applications which allow users to communicate with computers using natural language. CAs have been used effectively across many fields i.e. advice & guidance, customer services, computerized learning etc [21]. Among best known early developments were ALICE that was tasked to keep the users engaged in non-goal oriented conversations [22]. Other types of CAs developments include goal based Conversational Agents such as ADAM built to simulate a debt advisor for students [23] and OSCAR which was developed for predicting learning styles whilst simulating as a tutor [24].

IV. Existing Challenges faced by Natural Language Interface to Databases

The development of Natural Language Interfaces has been around for decades, but to date these systems are not in wider popular use. There are number of associated factors which discouraged the industry from adopting NLIDB systems [25]. Among notable challenges and problems identified by most NLIDB researchers are namely; linguistic problems, domain independence, translation process (i.e. database query formulation), multimodality and ease of configuration, conversational abilities etc [26, 2]. The problems such as grammatical utterances, anaphora and quantifier scoping are some of the linguistic problems that a NLIDB has to tackle when attempting to interpret a user utterance [27]. The lack of ability to elicit intended goals from inexperience users to retrieve information from a relational database, query results refinement and autonomous behavior remain among unsolved key challenges. In complex and distributed databases, many NLIDB systems have revealed ellipsis problems. Selection of conceptual models in constructing NLIDBs are also highlighted as one of the linguistic problems [28]. It is not ideal for users to remember what kind of question a given NLIDB system can or cannot answer. In the case where a NLIDB fails in understanding user requirements, it is often difficult for a user to judge the reason behind system failure (i.e. whether its scope of the system, system abilities or coverage of domain etc) [17]. Major problems confronting NLIDB in formulation of SQL queries have been defined as semantic ellipsis and wrong words or phrases (i.e. missing key information, adjectives, verbs and prepositions etc), coverage capabilities of SQL (i.e. several tables, aggregative functions, excessive information), user errors (i.e. request for information outside the database records or invalid utterances [27]. SQL query formulation issues are also associated with the adopted development approach [26]. Other challenging areas that are relative in hindering the advancement of NLIDB technology to develop an ultimate system can be described as namely; achieving high accuracy rates with domain independence, portability of knowledge domain and underlying database to work in different environment, ability to read meta knowledge (knowledge about knowledge), and explore big data in real time etc [2].

Additionally, more relevant to the proposed research, lack of conversational abilities in NLIDBs have been also outlined as a frequent problem by the users [18]. The abilities to converse naive users [7, 26, 29]. The development of NLIDB systems have been largely focused on single query response transactions. Little attention has been given to social adaptability, sustained interaction and pro-active conversation to elicit what an end user envisages about the domain, and how domain knowledge is structured [30]. Development of Natural Language Interfaces (NLIs) to retrieve information from structured data, requires understanding of Natural Language and complexities, ambiguities etc [31]. A number of architectural techniques have been adopted to use limited natural language to generate a logical query for structured data. Such techniques include Syntax-Based Family of Architectures, Semantic-Grammar Family of Architectures and Pattern Matching [5].

V. Knowledge Engineering the Domain

The knowledge engineering for the prototype system was carried out by researching complex SQL query architecting techniques and leading commercial business intelligence reporting systems used in different organisations. A number of sales reports were studied in the interest of understanding business process mapping. The structure of sales reports was also analysed in contrast with the common SQL query syntax used to generate information from the database. The research conducted on sales reports and complex query structures helped in constructing SQL query formulation features for the Aneesah system.

A relational database in a commercial environment often represents complex structure and burdened with millions of records. The records maintained in an organization's database are used for various purposes and can be classified into variety of layered information. The layered information can be further divided into master data, transaction data, enterprise information, and database objects/meta data [32]. A conventional knowledge engineering technique was adopted after a critical review and research on existing NLIDB systems and commercial enterprise reporting systems (i.e. Oracle, Microsoft, SAP, ASAP etc) to engineer the knowledgebase for the prototype system. This information modelling technique relies on capturing, representing, encoding and evaluation of expert knowledge (Chu, Hwang, Huang, and Wu, 2008). The prototype system has been equipped with expert knowledge with a generic approach in mind to allow naïve users to interactively retrieve and analyse information stored in chosen database for the developed system.

VI. ANEESAH NLIDB

Aneesah is a novel NLIDB system with conversational abilities to provide interactive and friendly environment to assist users with desired information stored in a relational database. Aneesah is able to correspond to the user requirements, guide the user to his/her intended goal and produce desired information from underlying relational database. The proposed architecture was developed by adopting the Pattern Matching (PM) approach.

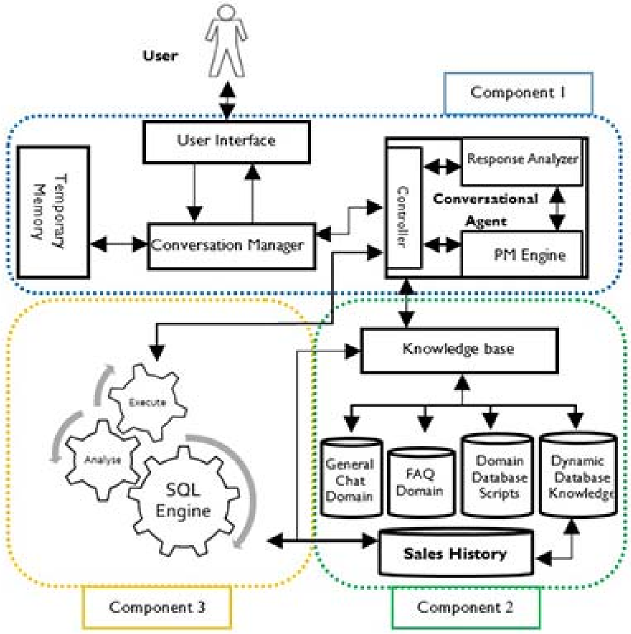

Fig 1. Aneesah NLIDB Architecture

Aneesah employs a novel framework constructed with a conversational agent based on scripting language, knowledge base and SQL engine. The developed prototype (Aneesah NLIDB) has been tested through a new evaluation methodology implemented using subjective and objective metrics to determine usefulness and effectiveness based on system experiments conducted following a pilot study. Figure 1 gives a high level overview of proposed architecture and role of components in supporting the Aneesah system's role to simulate as a human SQL developer by leading users in conversation to produce desired information from the relational database. A generic approach has been used to develop the proposed architecture, which is modular by nature for ease of development and maintenance with flexibility to integrate and adopt other domains with customization. The proposed framework comprises three major components as shown in Figure 1.

Component 1

Component 1 of the Aneesah system's architecture comprises of Conversation Manager, User Interface, and Temporary Memory and Conversational Agent components (Controller, Pattern Matching (PM) Engine, Scripting Language, and Response Analyzer). The Aneesah system features a novel Conversational Agent (CA) that is a fundamental part of the proposed architecture, which enables users to interact/converse with the Aneesah system for information retrieval purposes.

A. Controller

The controller module has been designed to handle direction of conversation with the users towards achieving desired information from the database. The controller leads conversation with the system users during interactive sessions. The controller also takes the responsibility of validating each user utterances before passing them to the PM engine for further processing in the system. An invalid utterance is a one off topic or not relevant to the domain, contains offensive or swearing words, or contains unnecessary characters. The controller features a built in three warnings rule in situations where it detects offensive words or abuse of the system. The user is shown with three warning messages detailing reason and error on their part, before system is closed by the controller. The validated utterances are forwarded to the PM engine by controller.

B. Pattern Matching Engine

The Aneesah system has been implemented using a novel PM engine. The PM engine controls user utterance matching against scripts in the system's knowledge base. The PM engine works based on a two tier approach (Tier 1 and Tier 2) approach. Tier 1 deals with user utterance matching against information stored in domain database (i.e. sales history database etc) tables to capture co-occurrence of attribute or key records leading to the formulation of a query based response, and Tier 2 deals with utterance matching against hand scripted patterns stored in Frequently Ask Questions (FAQ) and General Chat domains. The PM engine has been designed to work with rule based and non-rule based response handling. A rule based response can be described as a scripted textual response, executed following a successful utterance matching in either FAQ domain or General Chat domain. A non-rule based response relates to the formulation of a database query following successful utterance processing against Domain Database Scripts. The PM engine works on principle of pattern matching approach, which partially resembles an approach implemented in InfoChat system [4]. The PM engine works in conjunction with sentence similarity strength function to determine an appropriate response, and also interact with the user to resolve any ambiguity during conversation. The PM engine is equipped to deal with more than one response match situation. In the case, where a user utterance has attracted duplicate responses from different knowledge base domains, the PM engine uses sentence similarity calculation to execute the highest match value response. The user utterance is categorized once a match has been found in the knowledge base. The relevant domain (Domain Database Scripts, FAQ Domain or General Chat Domain) is activated once a user utterance is matched in the knowledge base. The domain activation is used by the PM engine to conversely engage with user and staying relevant to the conversing topic.

C. Pattern Matching Scripting Language

This section details the new scripting language approach adopted for the Aneesah system that allows users to engage in conversation with the system. The development of NLIDBs have been researched by number approaches, which represent associated challenges as discussed in literature review section (2.0) of this report. The pattern matching approach has been selected as the most suitable mechanism for developing a scripting language for the development of the Aneesah system. In order to model a novel scripting language for the Aneesah system, an extensive research was conducted on conversational systems such as InfoChat system [4].

D. User Interface

The Aneesah system is implemented with command line based user interface. The user interface is responsible for serving as an interaction platform between the user and Aneesah system. The user interface receives user utterances and displays back responses in the interface window. The information presenting technique has been implemented that has made it possible to display variable system responses in all forms i.e. text, quoted notions, query results etc.

Component 2

Component 2 of Aneesah's architecture consists of Knowledge base to serve as Aneesah's brain that has been developed based on the knowledge engineering techniques to specifically work with the domain database (Sales History database). The developed knowledge base can be customized with knowledge engineering to work with a different database. Responsible for supporting a two tier based user utterance matching across three different domains contexts. It is also responsible for providing domain knowledge to carry out user utterance mapping to match domain database information and support requests processing with minimum information from the users.

E. Knowledge base

The Knowledge base for Aneesah has been implemented in four layers of stored scripts namely; Domain Database Scripts, Frequently Asked Questions (FAQ) Domain, General Chat Domain, and Dynamic Database Knowledge (DDK). The Domain Database Scripts module plays fundamental role in prototype system by housing domain database relevant scripts. It is responsible for assisting PM engine to perform user utterance mapping against database information to capture database relevant information. FAQ contexts handles user questions related to the database structure. The General Chat domain handles user utterances outside database relevance for example user might want to talk about football, weather, etc. In this situation, the system attempts to motivate the user to stay relevant to the system's built purpose. The information maintained in sample sales history database, used for Aneesah's evaluation, is dynamically loaded into the DDK module on execution of system. The DDK module allows PM engine to perform matching of data records at all times and vacates the actual database from being reserved for match processes.

Component 3

Component 3 of the Aneesah architecture comprises novel SQL Query Engine with its components. The SQL Query Engine after having received database relevant syntax/information from controller component, is responsible for formulation of SQL queries in contrast with the users' utterances. The SQL engine identifies type and nature of queries following by activation of SQL configurator component.

F. SQL Query Engine and Components

The SQL Engine takes pivotal role in the Aneesah system's architecture. The SQL engine is responsible for performing dynamic query translation/formulation process from user utterances. The SQL engine works based on implemented techniques together with other sub components (SQL Configurator, SQL Execution, and SQL Analyser) in order to retrieve information from the database. The SQL engine relies on database relevant information delivered by the system's controller to analyse syntax requirements to engineer a query structure. The Aneesah system's query authoring knowledge extends users abilities to request information from any of 118 database columns in complex combination. The Aneesah system has been tested to support up to five tables joins. The table 1.1 illustrates the scope of SQL query formulation capabilities. The SQL engine utilises implemented query syntax to formulate database queries depending on the nature of users' requests.

The SQL Query engine is equipped with expert knowledge to identify level of complexity involved in a query formulation. In a real world scenario, a simple user request might lead to the need of variety of SQL query related syntax (i.e. database objects, aggregation functions, selection of results, filter etc) to satisfy his/her requirement. Therefore, the Aneesah system features an automated query structuring approach, which has the ability to gather necessary syntax to formulate query structures. The query formulation process adopts a step by step query syntax preparation process. The PM engine also initiates the SQL configurator to formulate a query. The query information is forwarded to the SQL configurator for further processing. The SQL configurator is responsible for generating syntactic information required to put together a database query structure. The formulated queries are transferred to SQL execution component, which takes the responsibility of executing and retrieving queries relevant results from the database. In addition to the query returned results, a scripted text response is also displayed to the user. The SQL analyser component has been implement as a failsafe tool, which takes the role of a database query structure analyser. Figure 2 shows the representation of query returned results in the user interface.

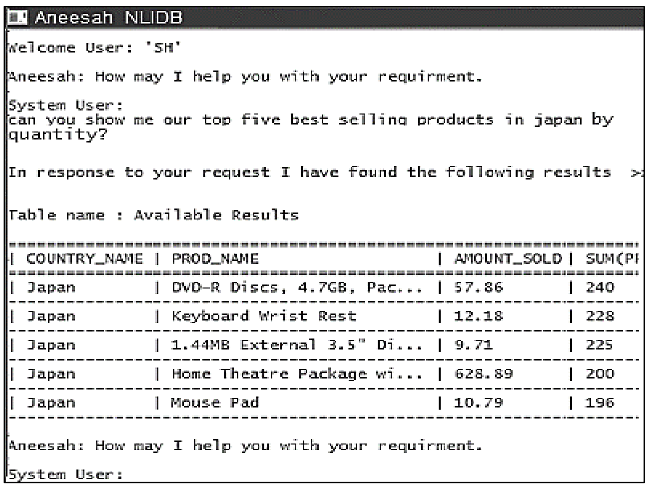

Fig 2: Example query returned results display in user interface

VII. Pilot Study

A. Domain

The domain database (Sales History database) used for prototype system's evaluation plays a fundamental role. It serves the main information retrieval source during

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

conversation sessions with the users. The chosen database comprises complex schema structure with records of large data related to Sales History (SH) records. The short listed SH schema for Aneesah system is particularly designed to demonstrate large amount of data in complex relational structure. The structure of the SH schema has been developed with sample sales records of electronic products belonging to an assumed company in view, which maintains a high volume operating business. The structure of SH database comprises 8 database tables with 114 columns containing domain relevant information.

B. Experimental Methodology

The evaluation methodology for the system required use of wider spectrum of evaluation (subjective & objective) metrics. The formulation of evaluation metrics was related to the major features of Aneesah system namely; dialogues naturalness, robustness and information accuracy etc. The developed system was evaluated through designed experiments, based on the formulated evaluation metrics. There were two experiments designed to provide the information required to carry out evaluation of Aneesah system. The information obtained from experiments was measured through statistical tests such as descriptive and inferential statistics. There were twenty participants selected to take part in evaluation. The participants were divided in two groups (Group-A & Group-B). The first Group-A comprised ten participants without possession of any databases or Structural Query Language (SQL) knowledge. The equal number of participants were selected for the second group (Group-B), who had familiarity with relational databases and SQL language either because of academic subjects or employment in similar field. The main purpose behind grouping participants in two equal groups was to take rigour system insights at initial testing stage. The null hypothesis (H0) was assumed that a general user cannot interact with a Natural Language Interface to Database to formulate a query and retrieve desired information from a relational database.

C. Results and Discussion

The questionnaire results for Experiment-1 from both participant groups (Group A & Group B) show that the Aneesah system has been well received. Overall 75% of participants from both groups rated the system interface (frontend) and level of understanding at high, however 30% of participant's rated these features between low and medium. Around 70% of the participants from both groups found the Aneesah system to be active, efficient and useful. The satisfaction level from Aneesah's dialog responses have been rated high by 70% of participants from both groups. Overall 90% of the participants agreed on using a similar system. The aim of Experiment 2 was designed to perform analysis such as robustness and accuracy. The information captured from the system's log file, which recorded occurrences of dialogs between participants and the Aneesah system during test sessions. For Group-A participants without possession of any databases or Structural Query Language (SQL) knowledge, the system has shown accuracy at 94.02% and precision was recorded at 90%. The evaluation of results performed for Group-B participants having familiarity with relational databases and SQL query language, measured less accuracy and precision comparing to the Group-A participants. The accuracy for Group-B participant is recorded at 80% and its precision is recorded at 87.50% for the same. In addition to the system's overall accuracy for both experimental groups (Group-A & Group-B) being scored at 85.01% and precision at 92.96%, the harmonic mean or F-measure equation is used to measure test of system accuracy by combining above accuracy and precision, which was recoded at 88.80%. Therefore, the null hypothesis (H0) is rejected and alternative hypothesis (H1) is assumed that, a general user can interact with a Natural Language Interface to Database to formulate a query and retrieve desired information from a relational database.

VIII. CONCLUSION AND FUTURE WORK

This paper presents a novel framework for the development of a conversation based NLIDB system. During evaluation, the proposed approach has shown its ability to allow users to take conversation based insights into the domain database. The development of Aneesah system was carried out by investigating existing approaches adopted for the development of similar systems. The aim of new approach described in this report, is to interpret users' requests by understanding and interacting with the users during conversation to produce required information from the database. The pattern matching approach solves the language complexities by simply matching user requests against scripted patterns and database information. Future work will be directed to evaluate Aneesah's abilities to handle more complex user requirements and to sustain dialogues and engage users in smart conversation to perform analysis of information by refining query results.

REFERENCES

[1] Elmasri, R., Navathe, S.B., 2010. Fundamentals of Database Systems, 6 edition. ed. Addison Wesley, Boston.

[2] Zelle, J.M., Mooney, R.J., 1996. Learning to Parse Database Queries Using Inductive Logic Programming, in: Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2, AAAI'96. AAAI Press, Portland, Oregon, pp. 1050–1055.

[3] Turing, A.M., 1950. Computing machinery and intelligence. Mind 433–460.

[4] O'Shea, J., Bandar, Z., Crockett, K., 2011. Systems Engineering and Conversational Agents, in: Tolk, A., Jain, L.C. (Eds.), Intelligence-Based Systems Engineering, Intelligent Systems Reference Library. Springer Berlin Heidelberg, pp. 201–232.

[5] William A. Woods, Robert M. Kaplan, and Bonnie Lynn Nash-Webber. The Lunar Sciences Natural Language Information System: Final Report. Technical Report BBN Report 2378, Bold Beranek and Newman Inc., Cambridge, Massachusetts, 1972.

[6] Woods, W. (1973). An experimental parsing system for transition network grammars. In Natural Language Processing, R. Rustin, Ed., Algorithmic Press, New York.

[7] Hendrix, G.G., Sacerdoti, E.D., Sagalowicz, D., Slocum, J., 1978. Developing a Natural Language to Complex Data. ACM Transactions on Database Systems 3, 105–147.

[8] Waltz, David (1975) Natural language access to a large database: an engineering approach. In Advance papers for the Fourth International Joint Conference on Artificial Intelligence (IJCAI-75), 868–872, Tbilisi, Georgia, USSR. – 8

[9] Waltz, D.L., 1978. An English Language Question Answering System for a Large Relational Database. Commun. ACM 21, 526–539. doi:10.1145/359545.359550

[10] David H. D. Warren and Fernando C. N. Pereira. An Ecient Easily Adaptable System for Interpreting Natural Language Queries. Computational Linguistics, 110-122. , July 1982. ISSN 0891-2017. URL http://portal.acm.org/citation.cfm?id=972942.972944

[11] Barbara J. Grosz, Douglas E. Appelt, Paul A. Martin, and Fernando C. N. Pereira. TEAM: An experiment in the design of transportable natural language interfaces. Artificial Intelligence, 32(2):173 – 243, 1987.

[12] Madeleine Bates (1989) "Rapid Porting Of the Parlance Natural Language Interface" In Proceedings of the Speech and Natural Language Workshop February 1989, Morgan Kaufmann Publishers, Inc., San Mateo, California, Pages 83-88.

[13] B.H. Thompson and F.B. Thompson. Introducing ASK, A Simple Knowledgeable System. In Proceedings of the 1st Conference on Applied Natural Language Processing, Santa Monica, California, pages 17–24, 1983.

[14] B.H. Thompson and F.B. Thompson. ASK is Transportable in Half a Dozen Ways. ACM Transactions on Office Information Systems, 3(2):185–203, April 1985.

[15] N. Ott, “Aspects of the Automatic Generation of SQL statements in a Natural Language Query Interface”, Information Systems, 17 (2), 1992, pp. 147-159.

[16] W. Sijtsma and O. Zweekhorst. Comparison and Review of Commercial Natural Language Interfaces. In F.M.G. de Jong and A. Nijholt, editors, Natural Language Interfaces, From Laboratory to Commercial and User Environments – Proceedings of the 5th Twente Workshop on Language Technology, Enschede, Twente University, NL, June 1993.

[17] Pazos, R.R.A., Rojas, P.J.C., Santaolaya, S.R., Martínez, F.J.A., Gonzalez, B.J.J., 2010. DialogueManager for a NLIDB for Solving the Semantic Ellipsis Problem in Query Formulation, in: Proceedings of the 14th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems: Part II, KES’10. Springer-Verlag, Berlin, Heidelberg, pp. 203–213.

[18] Pazos R, R.A., González B, J.J., Aguirre L, M.A., Martínez F, J.A., Fraire H, H.J., 2013. Natural language interfaces to databases: an analysis of the state of the art. Recent Advances on Hybrid Intelligent Systems 463–480.

[19] Raymond J. Mooney, Learning Language from Perceptual Context:A Challenge Problem for AI, American, Association for Artificial Intelligence (2006).

[20] Li, F., Jagadish, H.V., 2014. NaLIR: An Interactive Natural Language Interface for Querying Relational Databases, in: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, SIGMOD ’14. ACM, New York, NY, USA, pp. 709–712. doi:10.1145/2588555.2594519.

[21] P. Reis, N. Mamede, and J. Matias, “Edite – A Natural Language Interface to Databases: a New Dimension for an Old Approach”, Proceeding of the Fourth International Conference on Information and Communication Technology in Tourism, Edinburgh, Scotland, 1997.

[22] Wallace, R. S. (2008). ALICE: Artificial Intelligence Foundation Inc. Available at:http://www.alicebot.org.

[23] Pudner, K., Crockett, K.A., Bandar, Z., 2007. An Intelligent Conversational Agent Approach to Extracting Queries from Natural Language. In: World Congress on Engineering. pp. 305–310.

[24] Latham, K. Crockett, D. McLean and B. Edmonds (2010). Oscar: An Intelligent Conversational Agent Tutor to Estimate Learning Styles, 2010 IEEE International Conference, Barcelona.

[25] R, R.A.P., B, J.J.G., L, M.A.A., F, J.A.M., H, H.J.F., 2013. Natural Language Interfaces to Databases: An Analysis of the State of the Art, in: Castillo, O., Melin, P., Kacprzyk, J. (Eds.), Recent Advances on Hybrid Intelligent Systems, Studies in Computational Intelligence. Springer Berlin Heidelberg, pp. 463–480.

[26] Crockett, K., Bandar, Z, O’Shea, J and Mclean, D. (2009) ‘Bullying and Debt: Developing Novel Applications of Dialogue Systems’ In International Joint Conferences on Artificial Intelligence. Knowledge and Reasoning in Practical Dialogue Systems.

[27] Androutsopoulos, I., Ritchie, G., Thanisch, P.: Natural Language Interfaces to Databases - An Introduction. Natural Language Engineering 1, 29–81 (1995).

[28] H.R. Tennant, K.M. Ross, M. Saenz, C.W. Thompson, and J.R. Miller. Menu-Based Natural Language Understanding. In Proceedings of the 21st Annual Meeting of ACL, Cambridge, Massachusetts, pages 151–158, 1983.

[29] P.R. Cohen. The Role of Natural Language in a Multimodal Interface. Technical Note 514, Computer Dialogue Laboratory, SRI International, 1991.

[30] Owda, M., Bandar, Z., Crockett, K., 2007. Conversation-based natural language interface to relational databases, in: Web Intelligence and Intelligent Agent Technology Workshops, 2007 IEEE/WIC/ACM International Conferences On. pp. 363.

[31] Damljanovic, D., Agatonovic, M., Cunningham, H., 2012. FREyA: An Interactive Way of Querying Linked Data Using Natural Language, in: García-Castro, R., Fensel, D., Antoniou, G. (Eds.). Springer Berlin Heidelberg, pp. 125–138.

[32] Sapinsider.wispubs.com, (2015). Visualizing SAP: The Connected SAP Universe. [online] Available at: http://sapinsider.wispubs.com/Assets/Blogs/2011/October/Visualizing-SAP-The-Connected-SAP-Universe [Accessed 11 Jan. 2015].

[33] Docs.oracle.com, (2015). Sample Schema Scripts and Object Descriptions. [online] Available at: http://docs.oracle.com/cd/B28359_01/server.111/b28328/scripts.htm#Cihijfgh [Accessed 23 Mar. 2015].