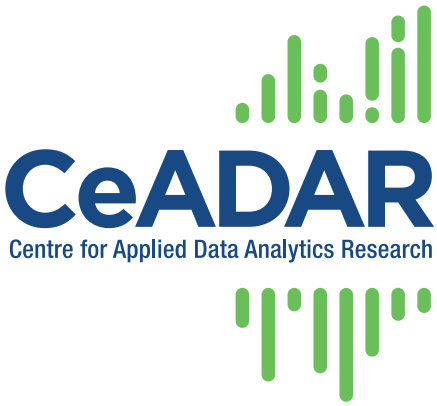

CeADAR – Centre for Applied Data Analytics Research

Enterprise Ireland Data Analytics Technology Centre

Voice Controlled Business Intelligence Systems - State of the Art Review

Document Type:	Literature Review
Project Title:	CeADAR
DDN Theme:	1 – Intelligent Analytic Interfaces
Theme Leader:	Brian Mac Namee (DIT)
DDN Sub-Theme:	Beyond the Desktop
Authors:	Eoghan O'Shea, Brian Mac Namee
Document Version:	1.0
Date of Delivery to ISG:	31st March, 2014
Number of pages:	16

ABSTRACT

In this review, we discuss the latest advances in *business intelligence* (BI) systems and, in particular, the latest advances in *voice controlled BI* systems. We discuss commercially available *voice recognition* software, and its limitations; some of the latest advances in the field of *natural language interfaces to databases* (NLIDB); and the related area of *clarification dialogues*. We conclude that there is a space in the market for a voice controlled business intelligence system, one that we believe can be robustly constructed using a restricted domain NLIDB system and leveraging a commercially available *off-the-shelf* voice recognition system.

CeADAR is a research partnerhip comprising University College Dublin, University College Cork, and Dublin Institute of Technology.

http://www.ceadar.ie

Contents

1 Introduction

[14] defines business intelligence as "*a set of methodologies, processes, architectures, and technologies that transform raw data into meaningful and useful information used to enable more effective strategic, tactical, and operational insights and decision-making*". The architecture of a typical enterprise-level BI system is shown in Fig. 1 (taken from [6]). It is worth noting that this is an abstract representation of a typical BI system and actual implementations can differ widely - including or excluding layers described below.

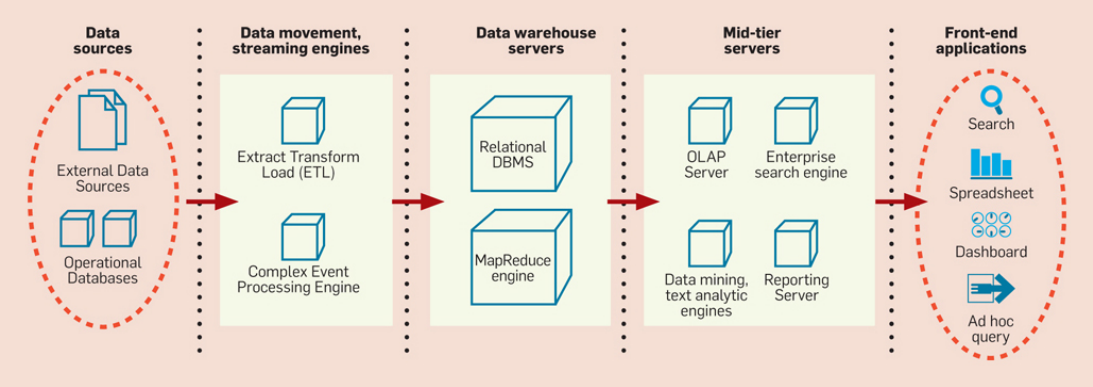

Figure 1: A typical business intelligence architecture (taken from [6])

As can be seen from Fig. 1, a large part of BI involves standardising and preparing data from initial data sources, which may contain both structured data (e.g. from SQL databases) and unstructured, or semi-structured, data (e.g. from emails, presentations). Extract-transform-load (ETL) tools in the first layer are used to perform the initial part of this process. Complex event processing (CEP) engines that can be used to infer if any meaningful events have occurred (e.g. a price exceeding a certain threshold value) are also often used in this layer. CEP engines are used in BI to provide near real-time processing and analysis of ongoing events based on live operational data itself.

In the second layer, data is typically loaded into a data warehouse managed by data warehouse servers. Storing and querying data in a data warehouse is typically handled by a relational database management systems (RDBMS). With the large increases in data being stored in this era of *big data*, systems based on MapReduce [18] are increasingly being used to aggregate data and store large data volumes more efficiently at this layer.

In addition to the Data Warehouse Servers there is typically a level of mid-tier servers in the third layer to carry out more specific functions. For example, online analytic processing (OLAP) servers which store data in multidimensional databases that enable a user to easily and selectively extract and compare data from many different angles and perspectives (e.g. products, geographic sales region, and time period would all be listed as separate dimensions, and from the OLAP server these entries could be cross-referenced). It is worth noting, however, that in many cases modern high-end hardware can allow this type of access to be made directly from raw data sources, making the use of OLAP redundant.

Other mid-tier servers include the reporting servers which enable definition, efficient execution and rendering of reports; and enterprise search engines which support the keyword

search paradigm over text and structured data in a warehouse (e.g. finding email messages, documents, history of purchases and support calls related to a particular customer). *Ad-hoc reporting* is a particularly important development at this level. Modern BI systems do not require all reports to be pre-defined but, rather, allow users to interact directly with their data defining queries on the fly which are answered immediately.

The final type of mid-tier servers are those related to specific data analysis tasks: data mining engines and text analytic engines. Data mining engines enable in-depth analysis of data that goes well beyond what is offered by OLAP or reporting servers, and provide the ability to build predictive models (e.g. to uncover which existing customers are likely to respond to any upcoming social media campaign). Text analytic engines can analyze large amounts of text data (e.g. survey responses or customer comments) and extract valuable information that would otherwise require significant manual effort [6].

In the final layer of a typical BI system, insight is communicated to the end-user, through some form of portal or front-end application (e.g. dashboards, spreadsheets or other ad-hoc visualisation tools).

There are a large number of commercially available BI software systems on the market today, including offerings from IBM, Oracle, Microsoft, SAS Tableau and Qliktech and Nathean (see [21] or [15] for good overviews). One of the most interesting recent developments in the design of these systems is that they are being targeted at non-technical users within organisations. Often referred to as *self-service BI* tools, these offerings typically focus on ad-hoc reporting and require interfaces that do not pre-suppose an understanding of technical topics such as statistics or query languages like SQL. Heavy use is made of data visualisations and *drag and drop* and *point and click* interfaces. Illustrative screenshots of a range of BI tools are shown in Fig. 2. Although these interfaces are useful in unlocking the power of BI to a wide range of users, they can become cumbersome and time consuming to use. In particular, to formulate queries users have to repeatedly populate templates which involves constant selecting from drop down lists.

One way in which to address this problem is to allow users to simply speak natural language queries to a system. This should make systems easier to use for the wide range of users that are now being targeted by self-service BI systems. This is also attractive in the mobile scenario in which the *drag and drop* and *point and click* paradigms are not as appropriate.

We note that the early commercial development of speech-to-query BI systems has been solely in the area of mobile BI. One of these systems is EasyAsk's Quiri [12] which, as their webpage puts it, "*eliminates the need to use awkward mobile keyboards or navigate hard-to-read menus*". The system is based on the Siri voice-recognition system built into Apple iPhones, coupled to a proprietary natural language engine and server (which is also used in the non-speech activated desktop system, Business Edition, offered by EasyAsk). It works by users tapping the microphone symbol on the iPhone and asking a question. Quiri quickly transcribes the question, converts it to a query, and retrieves an answer. Another and similar mobile BI system is SAVANT Mobile BI for Essbase ([4]) which leverages the Nuance speech recognition system to extract financial information from Essbase's OLAP databases.

Figure 2: Typical Business Intelligence Dashboards. Images from [21] and [5].

It is this type of system that is the focus of this report. Our general interest is in the improved functioning of data analytics interfaces and, in particular, improving the accessibility of self-service BI systems through the use of voice interfaces which allow queries to be stated in spoken, natural language rather than using query languages are cumbersome interface elements. We consider that to build a voice-controlled BI system there are three main tasks that must be addressed:

(i) Speech recognition
(ii) Translation from natural language into structured queries
(iii) Clarification dialogues.

We will discuss each of these components separately in the sections that follow below.

2 Speech Recognition

The field of speech recognition is a well developed one [3]. While the first attempts at solving this technical challenge took place as far back as the 1920s, the most significant developments have occurred since the 1960s. More recently, speech recognition has improved significantly with the use of hidden Markov model systems, augmented by artificial neural networks [39]. In this work, it is not our intention to develop our own speech recognition system, rather it is our intention to use a reliable off-the-shelf system for purchase/license.

There are a number of standalone commercial speech recognition software systems on the market, including Dragon Naturally Speaking Premium, TalkTyper, Tazti and TalkingDesktop. See [31] and [20] for more information on these commercial software systems. Another

well-known proprietorial speech recognition software suite is Vox Sigma, developed by Vocapia Research ([43]). Embedded speech recognition software is included as part of Windows (Windows Speech Recognition) and Google (Google Voice Search) operating systems. Using APIs made available by Microsoft (Microsoft Speech API) and Google (Web Speech API), it is possible to incorporate these speech recognition systems into any application. We note that an open source alternative, CMU Sphinx, is also available ([40]). Dragon Naturally Speaking Premium is considered to be the market leader in this area.

The most common metric used to evaluate the performance of speech recognition systems is *word error rate* (WER). This measures the percentage of words mis-recognised in a text speech segment. for modern speech recognition systems WERs of approx. 20–30% for telephone-based conversations are typical ([32]). Lower WER values of approx 15% are found when a subject reads in a text based corpus, such as the WSJ lexicon ([32]). Nilsson ([30]) examined a number of different speech recognition systems (Microsoft SAPI, Pocket Sphinx and Dragon Naturally Speaking), and their effectiveness in recognising speech from videos, finding WERs of 42–61% (reduced to 39.5% for Pocket Sphinx after modifying the language model and dictionary).

Vertanen ([42],[41]), reading in the WSJ lexicon, found lower WERs of between 24–39% (with Dragon Naturally Speaking having the lowest WER of 23.9%). Lower WERs would be expected in this case, compared to the work of Nilsson on videos ([30]), due to the words being read into the system, e.g. Pallett ([32]) records that WERs of approx 15% are possible in these types of scenarios. The WERs of speech recognition systems built into commercially available systems, e.g. Siri on the iPhone 4 or Microsoft Windows Speech Recognition system (available in Windows 7) have, in comparison, WERs of 12% (Windows) and 18% (Siri), based on a dictated text of 604 words ([16]).

The WERs of all systems reviewed here are above 15%. However, it should be noted that in any voice-controlled BI system, it is probable that very short commands will be given in a very limited domain (e.g. the commercial area of the particular business). It is shown in Pallett ([32]) that in a similarly restricted domain (e.g. air travel information systems), WERs of approx. 2.5% are more typical. WERs in this range are in the same range of human error in transcription, i.e., they are equivalent to mistakes a human listener might make on hearing the same text being spoken.

3 Translation from Natural Language into Structured Queries

Natural language interfaces to databases (NLIDB) are systems that allow a user, through the medium of spoken or typed natural language, to obtain information stored in a database [23], [2]. It is not our intention here to give a full review of the field of NLIDB (we refer the reader to reviews such as [2] or [29] for this) but would like to discuss some of the latest developments in this area. Before doing this, we will discuss briefly the different possible architectures of NLIDBs and the issue of portability.

In terms of architectures, there are considered to be four main types of NLIDB:

- **Pattern matching systems** are systems that just take a sentence (a user's question) to be an arbitrary sequence of tokens and look for the presence of some given pattern within

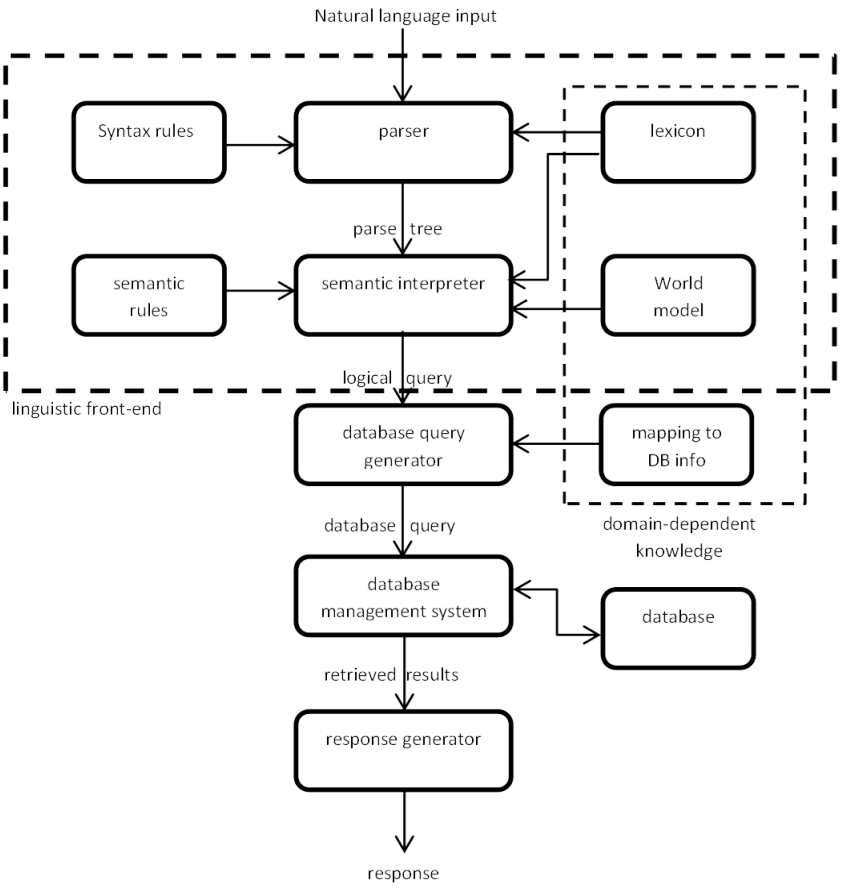

Figure 3: Possible architecture of intermediate representation language system. Image based on Fig. 5 from [2].

these tokens

- **Syntax-based systems** where the user's questions are analysed syntactically and the resulting parse tree is directly mapped to an expression in a database query language [2]
- **Semantic grammar systems** which are very similar to a syntax-based systems, except that the parse trees are simplified as much as possible by removing unnecessary nodes or combining nodes together [29];
- **Intermediate representation language systems** where the natural language input is first processed syntactically by a parser and then the resulting parse tree is transformed to an intermediate logic query using a semantic grammar system, taking into account words stored in a lexicon and a "world model" that typically contains a hierarchy of classes of world objects (e.g. positions in a company), and constraints on the types of arguments each logic predicate may have, before translation to the database's query language. The architecture of a typical intermediate representation language system is shown in Fig. 3.

Following [23], we can further distinguish NLIDBs as either **domain dependent** or **domain independent**. Domain dependent NLIDBs are designed to be used only in a specific domain and use knowledge about the domain to add the context required to handle user queries. Early NLIDBs were very much of this type, and were designed for specific databases, e.g.

LUNAR [2]. More recent work in domain-dependent NLIDBs has developed reconfigurable systems in which the domain knowledge required for different domains can be added by a trained technical user, e.g. ASK [2]. There have been some **auto-reconfigurable** NLIDBs developed that can automatically acquire the domain knowledge needed through interactions with a user. NLIDBs of this type typically use an intermediate representation language, e.g. see [23]. Examples of this category include AskMe* [23], GINLIDB [11], FREyA [8] and the NLIDDB for the CINDI virtual library [36].

In domain-independent NLIDBs, no knowledge is stored about the underlying domain, and natural language queries are translated directly into SQL queries and executed against the underlying database. However, since the system does not know anything about the domain, it is not able to warn the user about any potential conceptual errors in the query. The PRECISE system [33] is a good example of a domain-independent NLIDB.

In terms of recently developed, state-of-the-art, systems in NLIDB, we will discuss here AskMe* [23], GINLIDB [11], FREyA [8], NLIDB for CINDI virtual library [36], PRECISE [34], [33] and WASP [44], [45].

3.1 AskMe*

AskMe* is a database independent NLIDB that uses an intermediate representation language system to construct queries, making use of an ontology to store domain entities, properties, relationships and constraints. It makes use of the link grammer parser [37]. AskMe* provides a query-authoring service providing query suggestions (by providing drop down pop-up menus with words contained in the lexicon), syntax error highlighting and domain-specific error handling (i.e. making sure that lexically and syntactically valid queries are also valid semantically). Queries that are not valid are notified to the user in a visual way (i.e. the portions of the query causing the inconsistency are highlighted and a tooltip contains a description of the inconsistency if the mouse hovers over the highlighted regions).

3.2 GINLIDB

GINLIDB is similar to AskMe*, being also an intermediate representation language system, although it makes use of a probabilistic context free grammar (PCFG) parser. The main difference is its use of a graphical user interface (GUI) as the portal by which a user interacts with the system, i.e. by which they enter their (typed) natural language query. By means of the GUI, GINLIDB provides a spell checker to correct for any misspelt queries, allows words to be added to the existing knowledge base, has a database mapping procedure to associate newly added words with synonyms (and to the appropriate attribute and database) and has a database relationship handler to establish relationships between the different tables in the database [13].

3.3 FREyA

FREyA [8] is again similar to AskMe*. It uses the knowledge encoded in ontologies as the primary source for understanding the user's question and only then tries to use the output

of the syntactic parsing in order to provide the most precise answer. If the system is not able to automatically derive an answer, it will generate clarification dialogues. From these clarification dialogues, the user's choice is saved and used for training the system in order to improve its performance over time.

3.4 The NLIDB for the CINDI Virtual Library

The NLIDB for the CINDI virtual library [36] is designed to be domain independent, and shares similarities with AksMe*, FREyA and GINLIDB. In this system, input sentences are syntactically parsed using the link grammar parser [37] and then semantically parsed through the use of domain-specific templates. The system is composed of a pre-processor and a run-time module. The pre-processor builds a conceptual knowledge base from the database schema using WordNet [26]. This knowledge base is then used at run-time to semantically parse the input and create the corresponding SQL query.

3.5 PRECISE

PRECISE [33] is a system where the target database is in the form of a relational database using SQL as the query language. The system combines the idea of a plug-in parser (allowing it to be subsequently changed to take advantage of advances in this area) and the concept of semantic tractability, i.e. sentences that can be translated to a unique semantic interpretation by analysing some lexicons and semantic constraints [29]. The strength of PRECISE is based on its ability to match tokens in a sentence to the corresponding database structures by firstly narrowing the possibilities using a Maxflow algorithm, before passing any possible SQL queries to an "*quivalence checker*" [34].

3.6 WASP

WASP (Word Alignment-based Semantic Parsing) [44], [45] is a novel statistical approach to semantic parsing (the results of which can then be applied to a NLIDB domain). The WASP algorithm learns a semantic parser given as a corpus a set of natural language sentences annotated with their correct meaning representations (e.g. formal query language translations). It uses a statistical word alignment model to acquire a bilingual lexicon consisting of natural language substrings coupled with their translations in the target machine representation language (query language). The system works in two parts, firstly a lexicon is created by finding the word alignments between natural language sentences and their correct meaning representations, and then, secondly, a probabilistic model is used to derive the most probable derivation.

3.7 The High-level Query Formulator

The High-level Query Formulator [25] relies on a graph representation of the semantic model of the underlying database (the "*semantic graph*") to formulate valid SQL queries. The system accepts user input in natural language and, using keywords approximates the meaning of a user's input. Keywords are words or phrases within the database domain that have a particular meaning. Nodes and attributes of the semantic graph, the values contained in the

database and operators are all keywords in the system. From the user's natural language input, the system determines what keywords are present and how they correspond to the components of the Semantic graph, database values or operators. Because keywords may have several meanings within a given domain, a statistical method based on comparing n-gram vectors is used to disambiguate keyword meanings. The final meaning of the keywords from the user's input are used by the High-level Query Formulator to compose a formal database query. The advantages of this system are that it uses no grammars or much outside knowledge and most of the knowledge of the system can be obtained from the database itself; it is therefore extremely portable. As [25] mention, the system is most suitable for applications where the user input is short and simple, and where the domain has a limited ontology.

4 Clarification Dialogues

In an NLIDB it is natural to expect that some interactions will break down, e.g. if requests are not properly formed by the user or because some issue arises in the understanding of the request by the system. In these circumstances, a clarification dialogue can be used to correct misunderstandings, under-specifications, spelling mistakes, etc. There has been a large amount work done in the field of spoken dialogue systems, specifically in the field of clarification dialogues, e.g. [17], [22], [10]. Further work in attempting to determine a user's goals under uncertainty, and thereby compute the next optimal system action (e.g. ask for clarification or offer an alternative) has taken place in research in statistical spoken dialogue systems through the application of partially observable markov decision process (POMDP) approaches ([7], [38]).

[9] investigated two methods for improving the usability of a natural language interface: feedback and clarification dialogues. Feedback shows the user how the system interprets the query, suggesting repair through query reformulation. Clarification dialogues are developed which offer suggestions, found through ontology reasoning, to the user. These clarification dialogues are coupled to a learning mechanism. However, we note that [9] considers purely a text-based natural language system (i.e. no spoken component) and that [23] also uses a form of clarification in their system, AskMe*, with spell checks, etc. Other similar sytsems are those of [24](Aqualog) and [19] (Querix).

5 Restricted domains: A possible solution

One solution to the problem of building natural language interfaces to databases is to use restricted domains ([27]). Minock ([27]) considers that there are certain characteristics that a restricted domain should have:

- it must be circumscribed, with a clearly understood (to the user) domain area and available level of detail
- it must be complex, i.e. sufficiently broad in terms of structure, containing numerous concepts (within the circumscribed domain) and entities that have complex properties associated with them
- it must be practical, i.e., it should contain information of use to a wide range of people, be feasible to build and maintain for a particular domain, and the effort of constructing it must be justified by its use (quantity of queries) over the lifetime of the interface.

Minock ([27]) considers plausible restricted domains to be things such as bus schedules, city events and information and software catalogues. We consider here that other valid restricted domains are ones related to particular business activities and, as such, domains that are highly suitable for inclusion in a BI system.

It is not our intention here to review the history of research into restricted domains in full, we refer the reader to [28] for a review of question answering in restricted domains. Here, we will discuss in detail one illustrative example a specific recent implementation of a restricted domain NLIDB system, that of Agrawal & Kakde ([1]).

This work of Agrawal & Kakde ([1]) describes a method for the semantic analysis of natural language queries to databases using a domain ontology. The work is carried out using a restricted domain based on Indian railway information. In this work, a domain ontology structure is first constructed, collected from railway guides, magazines and websites; this ontology has three major concept types:

- Objects: static things that exist in the real world, e.g. Train, Department, etc.
- Events: any activities, actions, happenings that occur over some period of time
- Properties: describe properties of Objects and Events (e.g. Express is a Property of the Object Train, as in an Express Train)

Their system uses deep semantic analysis in which the input query is analysed at word and sentence level using detailed language knowledge. They consider that a query has two important constituents: what is expected (the main subject of the query) and the constraints on this. Their system has a number of stages. In the first stage, the natural language query is preprocessed, undergoing part of speech (POS) tagging, chunking (finding the base syntactic constituents) and named entity recognition (NER). They use the NLTK with Python for this preprocessing.

The next stage is to pass the results of this preprocessing to a language modeller. In this stage, the idea is to identify the most important named entity (the "expected" entity) among the named entities identified in the query. This is done by examining the words in the query, e.g. in queries starting with the word "What" the noun phrase immediately after the wh-word determines the expected entity. Similarly, if the query starts with "When" the verb of the sentence determines the expected entity. A rule base is generated using such rules for a wide variety of questions. This rule base system has the advantage of being domain independent.

The final stage of the system employs ontology mapping. The "expected" entity and other named entities (constraints) from the named entity recogniser are mapped to ontology terms (Objects, Events, Property). This mapping from the natural language entities to ontology terms is done by using a Lexicon. This Lexicon is populated using a training corpus of domain specific natural language queries, and contains linguistic phrases (corresponding to all possible Named Entities in the restricted domain) along with their respective ontology term and property. A Named Entity is matched to a phrase in the Lexicon using a Jaccard similarity measure, and the corresponding ontology term (for the more similar phrase) is extracted. The output from these stages are the expected entity and the constraints on this (the two important constituents of the query), from which a structured query to the database can be constructed.

This system was tested on a number of casual users, not involved in the system design, and in most cases was found to return correct answers with a precision of 80-90%. We see from this example that a workable system can be constructed from a restricted domain, one that can return results with a high degree of accuracy. We consider that restricting the domain, and employing a system like that of Agrawal & Kakde ([1]), is one valid solution to the problem of constructing a natural language interface to the database of typical BI system (we assume that a BI system works off a reasonably restricted domain, i.e. one related solely to commonly occurring business matters, business area, etc.).

6 Conclusions

Our intention in this report was to review the state of the art in the field of voice-controlled BI systems. We conclude that while there are numerous commercially available BI software systems on the market, there are few commercial examples of voice-control being used. Those that exist at present (as far as the authors are aware) are exclusively in the area of mobile BI, i.e. EasyAsk Quiri and SAVANT Mobile BI for Essbase. These mobile devices leverage powerful voice recognition systems, Siri (the iPhone ASR system) for Quiri and the Nuance ASR system for the SAVANT system. The lack of a desk-top voice controlled BI systems is puzzling, but is perhaps related to the difficulties of producing a natural language interface to a database (NLIDB), something that is still a large area of ongoing research (as we have partially detailed in this report). This is expected to be an emerging trend in future BI systems, however. For example, Gartner highlight voice-controlled BI as an upcoming hot topic [35].

We consider one solution to the problem of constructing a robust NLIDB is to use a restricted domain, e.g. as used in a recent paper by Agrawal & Kakde ([1]). We consider that this would tie in well with the type of restricted domains typically present in most business environments, i.e., businesses that deal with one market segment only such as car sales, hardware, etc. We do not consider that the relative failings of differing voice recognition systems to be relevant to this project, with the type of short commands expected to be used in a voice-controlled BI system having WERs of approx. 2.5%, i.e. equivalent to the error rate of a human listener.

We can conclude from this report that there is a space in the market for a voice-controlled BI systems, one that we believe can be robustly constructed using a restricted domain NLIDB system, leveraging one of the commercially available voice recognition systems, e.g. Dragon Naturally Speaking, etc.

References

[1] A.J. Agrawal and O.G. Kakde. Semantic analysis of natural language queries using domain ontology for information access from database. *International Journal of Intelligent Systems & Applications*, 5(12), 2013.

[2] L. Androutsopoulos. Natural language interfaces to databases - an introduction. *Journal of Natural Language Engineering*, 1:29–81, 1995.

[3] M.A. Anusuya and S.K. Katti. Speech recognition by machine, a review. *International Journal of Computer Science and Information Security, IJCSIS*, 6.3:181–205, 2009.

[4] Hyperion Architects. Savant mobile bi for essbase, 2014. Accessed March 2014 (http://hyperionarchitects.com/mobile-bi-for-essbase/).

[5] Chris Boylan. Infoapps (http://www.informationbuilders.com/info-apps-infoapps). Technical report, February 2014.

[6] S. Chaudhuri, U. Dayal, and V. Narasayya. An overview of business intelligence technology. *Communications of the ACM*, 54.8:88–98, 2011.

[7] P.A. Crook and O. Lemon. Representing uncertainty about complex user goals in statistical dialogue systems. In *Proceedings of SIGDIAL 2010: the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 209–212, 2010.

[8] D. Damljanovic, M. Agatonovic, and H. Cunningham. Natural language interfaces to ontologies: combining syntactic analysis and ontology-based lookup through the user interaction. *Proceedings of the 7th Extended Semantic Web Conference*, pages 106–120, 2010.

[9] Danica Damljanovic, Milan Agatonovic, Hamish Cunningham, and Kalina Bontcheva. Improving habitability of natural language interfaces for querying ontologies with feedback and clarifcation dialogues. *Web Semantics: Science, Services and Agents on the World Wide Web*, 19(0), 2013.

[10] Marco De Boni and Suresh Manandhar. An analysis of clarification dialogue for question answering. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–55, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[11] P.R. Devale and A. Deshpande. Probabilistic context free grammar: an approach to generic interactive natural language interfaces to databases. *Journal of Information, Knowledge and Research in Computer Engineering*, 1(2):52–58, 2010.

[12] EasyAsk. Quiri, 2014. Accessed March 2014 (http://www.easyask.com/products/quiri/).

[13] F. A. El-Mouadib, Z. S. Zubi, A. A. Almagrous, and I. El-Fegh. Generic interactive natural language interface to databases (ginlidb). *Proceedings of the 10th WSEAS International Conference on Evolutionary Computing*, EC'09:57–76, 2009.

[14] B. Evelson. Topic overview: Business intelligence - an information workplace report. Technical report, www.forrester.com, 2008. http://tinyurl.com/8r642jn.

[15] Boris Evelson. The forrester wave: Enterprise business intelligence platforms, q4 2013. Technical report, Forrester Research, 2013.

[16] L. Fradrich and D. Anastasiou. Siri vs. windows speech recognition. *Translation Journal*, 16(3), 2012. http://www.bokorlang.com/journal/61dictating.htm (accessed March 2014).

[17] M. Gabsdil. Clarification in spoken dialogue systems. *Proceedings of the 2003 AAAI Spring Symposium. Workshop on Natural Language Generation in Spoken and Written Dialogue*, AAAI Technical Report SS-03-06, 2003.

[18] S. G. Jeffrey Dean. Mapreduce: Simplified data processing on large clusters (http://research.google.com/archive/mapreduce.html). Technical report, Google Research Publications, December 2004.

[19] Esther Kaufmann, Abraham Bernstein, and Renato Zumstein. Querix: A natural language interface to query ontologies based on clarification dialogs. In *In: 5th ISWC*, pages 980–981. Springer, 2006.

[20] Katie Keipp. 2014 compare best voice recognition software (http://voice-recognition-software-review.toptenreviews.com/). Technical report, TopTenReviews.com, 2014.

[21] Michael Koploy. Compare business intelligence (bi) software tools. Technical report, February 2014.

[22] Holzapfel H. Waibel A. Krum, U. Clarification questions to improve dialogue flow and speech recognition in spoken dialogue systems. In *Interspeech '05*, pages 3417–3420, 2005.

[23] M. Llopis and A. Ferrandez. How to make a natural language interface to query databases accessible to everyone: An example. *Computer Standards & Interfaces*, 35:470–481, 2013.

[24] Vanessa Lopez, Victoria Uren, Enrico Motta, and Michele Pasin. Aqualog: An ontology-driven question answering system for organizational semantic intranets. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):72 – 105, 2007. Software Engineering and the Semantic Web.

[25] Frank Meng and Wesley W. Chu. Database query formation from natural language using semantic modeling and statistical keyword meaning disambiguation. Technical report, Computer Science Department, University of California, Los Angeles, 1999.

[26] George A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[27] Michael Minock. Where are the 'killer applications' of restricted domain question answering. In *Proceedings of the IJCAI Workshop on Knowledge Reasoning in Question Answering*, page 4, 2005.

[28] Diego Mollá and José Luis Vicedo. Question answering in restricted domains: An overview. *Computational Linguistics*, 33(1):41–61, 2007.

[29] N. Nihalani, S. Silakari, and M. Motwani. Natural language interface for database: a brief review. *International Journal of Computer Science*, 8(2):600–608, 2011.

[30] Tobias Nilsson. Speech recognition software and vidispine. Master's thesis, Umeï£¡ University, Department of Computing Science,, 2013.

[31] Mark O'Neill. Control your pc with these 5 speech recognition programs (http://www.pcworld.com/article/2055599/control-your-pc-with-these-5-speech-recognition-programs.html). Technical report, http://www.pcworld.com, November 2013.

[32] David S Pallett. A look at nist's benchmark asr tests: past, present, and future. In *Automatic Speech Recognition and Understanding, 2003. ASRU'03. 2003 IEEE Workshop on*, pages 483–488. IEEE, 2003.

[33] A. Popescu, A. Armanasu, O. Etzioni, D. Ko, and A. Yates. Precise on atis: Semantic tractability and experimental results. *Proceedings of the National Conference on Artificial Intelligence*, AAAI:1026–1032, 2004.

[34] Ana-Maria Popescu, Oren Etzioni, and Henry Kautz. Towards a theory of natural language interfaces to databases. In *Proceedings of the 8th international conference on Intelligent user interfaces*, pages 149–157. ACM, 2003.

[35] Gartner Research. Gartner says business intelligence and analytics need to scale up to support explosive growth in data sources. Technical report, 2014.

[36] N. Stratica, L. Kosseim, and B.C. Desai. Using semantic templates for a natural language interface to the cindi virtual library. *Data and Knowledge Engineering journal*, 55(1):4–19, 2004.

[37] D. Temperley, J. Lafferty, and D. Sleator. Link grammar parser, accessed Feb. 2014. http://www.abisource.com/projects/link-grammar/.

[38] B. Thomson and S. Young. Bayesian update of dialogue state: A pomdp framework for spoken dialogue systems. *Computer Speech and Language*, 24(4):562–588, 2010.

[39] Edmondo Trentin and Marco Gori. A survey of hybrid ann/hmm models for automatic speech recognition. *Neurocomputing*, 37(1):91–126, 2001.

[40] Carnegie Mellon University. Cmusphinx, 2014. Accessed March 2014 (http://cmusphinx.sourceforge.net/wiki/).

[41] Keith Vertanen. Baseline WSJ acoustic models for HTK and Sphinx: Training recipes and recognition experiments. Technical report, Cavendish Laboratory, University of Cambridge, 2006.

[42] Keith Vertanen. Speech and speech recognition during dictation corrections. In *Proceedings of the International Conference on Spoken Language Processing*, pages 1890–1893, September 2006.

[43] Vocapia. Voxsigma, 2014. Accessed March 2014 (http://www.vocapia.com/).

[44] Yuk Wah Wong. *Learning for semantic parsing using statistical machine translation techniques*. Computer Science Department, University of Texas at Austin, 2005.

[45] Y.W. Wong and R.J. Mooney. Learning for semantic parsing with statistical machine translation. *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 439–446, 2006.