LRRP SpeechWebs

Richard Frost, Nabil Abdullah, Kunal Bhatia, Sanjay
Chitte, Fadi Hanna, Maxim Roy, Yue Shi and Li Su.
School of Computer Science, University of Windsor, Ontario, Canada
richard@uwindsor.ca

Abstract

This paper describes a new architecture for accessing hyperlinked speech-accessible knowledge sources that are distributed over the Internet. It differs from the use of speech interfaces to conventional web html pages; from conventional telephone access to remote speech applications (as used in many call centers); and from the use of a network of hyperlinked VXML pages. The architecture is ideally suited for use when cellphones become available with built-in speech-to-text and text-to-speech capabilities.

Keywords: SpeechWeb, hyperlinked speech applications, distributed speech recognition.

1. Introduction

Dale Hartzell, V.P. Sandcherry Inc., has quoted a Kelsey Group study which estimates that speech-enabled services will generate over $4.6 billion in service revenue for North American wireless carriers, and $25 billion worldwide, by 2006. Hartzell claims that this will only become a reality if speech solutions can deliver flexibility, scalability and economy [10], and can be deployed as easily as web services can be deployed [11]. It is our belief that these requirements can only be met if currently-used distributed-speech architectures are augmented with a new architecture that we shall refer to as an LRRP SpeechWeb architecture.

An LRRP SpeechWeb uses **L**ocal thin-client application-specific speech **R**ecognition and **R**emote natural-language query **P**rocessing. Users navigate an LRRP SpeechWeb using voice-activated hyperlink commands, and query the knowledge sources through spoken natural-language using a speech browser executing on a local device [4]. Natural-language speech input is required as it is difficult for end-users to express queries verbally in a formal query language such as SQL.

Currently, the preferred speech-recognition technology for LRRP SpeechWeb browsers is grammar based, rather than statistical, for two reasons: 1) large corpuses are needed to build statistical speech models and such data is not available for many of the user-built knowledge sources, and 2) readily-available commercial speech-recognition technology that can be executed on lightweight end-user devices is grammar-based.

When an LRRP SpeechWeb browser is directed to a remote hyperlinked knowledge source (which we shall refer to as a "sihlo" from now on) it begins by downloading a recognition grammar which is used to tailor the browser to the knowledge contained in the remote sihlo. Queries that are subsequently recognized on the local device are then sent to the remote sihlo for processing. Answers that are returned to the local browser are output as synthesized voice. If a navigation command is spoken, the sihlo returns the address of another sihlo to which the browser is redirected in a manner similar to the following of a link in the conventional web.

Grammars can be simple or complex, and sihlos can be constructed in any programming language. In the most-simple case, the grammar might consist of a list of questions that can be asked; and the natural-language query processor could be coded in a scripting language as a set of "canned" answers of the form "if the input is "what is your name" the output is "Joe Smith"". Service providers with more advanced programming knowledge can construct more sophisticated sihlos

1.1 How LRRP SWs differ from speech interfaces to the conventional web

Speech interfaces that are used to interact with conventional html pages have been available for several years. Early versions allowed users to scan html pages using voice commands and output text in synthesized voice. More recent products translate conventional web

pages into speech dialogues usually coded in VXML (see below). Such interfaces are useful in some applications, but have shortcomings which result from the fact that html pages are designed for visual browsing, and users cannot ask database-query-like questions about html-page contents. In addition, key-word and phrase-matching techniques that are used by web search engines are not appropriate for speech access as the range of possible words and phrases is so large as to preclude acceptable speech-recognition accuracy.

1.2 How LRRP SWs differ from telephone access to remote VXML applications

Call centers allow users to phone in and speak to applications running on their machines. Speech recognition is carried out at the call centre. The emerging standard for implementing the remote applications is the voice-application markup language VXML [14]. VXML is much like html except that it includes commands for prompting user speech input, for invoking grammars for recognizing input, for outputting synthesized voice, for iteration through blocks of code, for calling ECMA scripts or Java objects, and for calling remote VXML pages that are downloaded and executed by the VXML interpreter in a similar way as html pages are linked in the conventional web.

A relatively-recent development is to integrate the call-centre model with 1) software that converts html pages to VXML dialogues as discussed in 1.1 above, and 2) software that allows VXML pages to be stored on conventional web servers and be accessed through remote telephones [18]. We shall refer to the call-centre architecture and its extensions as the Remote Recognition/Remote Processing (RRRP) SpeechWeb architecture. The RRRP architecture is finding numerous applications in industry and is likely to see huge growth over the next few years. However, it has three shortcomings: 1) For complex applications, the speech-recognizer needs to be trained for the individual user in order to achieve the required level of accuracy. With the RRRP architecture, user profiles that are developed through training would have to be stored at each call centre. This is clearly not practical if a large SpeechWeb is to be established. 2) With the RRRP architecture, all speech recognition is carried out at the service-provider site placing a huge computational burden on the provider's machines for applications with complex input languages, 3) As noted by Herzell deploying speech services should be as easy as deploying regular web services. In the RRRP architecture, service providers have to embed their applications in VXML, provide speech-recognition capabilities, maintain user voice-profiles to improve accuracy, and employ specialized software to make speech applications available over the web. In contrast, our proposed LRRP architecture requires only that the provider maintain a recognition grammar in a standard format (eg the Java Speech Grammar Format-JSGF) and a small script to make their applications available through a text-in/text-out web protocol such as CGI (see later). The grammar, script, and application can all be placed in a regular web directory alongside html pages.

1.3 How LRRP SWs differ from sets of downloadable hyperlinked VXML pages

A voice application can be constructed as a set of hyperlinked VXML pages [14]. Such pages can be downloaded to VXML interpreters executing on client machines which perform functions locally as directed by the VXML page, including answering queries possibly involving access to remote sources. We refer to such use of VXML as a Local Recognition/Local Processing (LRLP) SpeechWeb architecture. The LRLP architecture is appropriate for many applications but has shortcomings for others: 1) natural-language query interfaces and their associated knowledge sources can be very large and are best executed at remote sites if the end-user has a lightweight device, 2) providers should be able to use whatever programming language they like and not have to embed their applications in VXML for the reasons discussed above.

1.4 Potential use of LRRP SpeechWebs and problems to be solved

The LRRP architecture can be integrated with speech interfaces to the conventional web, with call-centre style architectures, and with collections of VXML pages. The architecture is ideally suited for general-purpose speech access to distributed hyperlinked knowledge sources, especially when lightweight end-user devices, such as J2ME/VXML-enabled cellphones become available. LRRP SpeechWeb browsers and their recognizers can be tailored for individual users thereby significantly improving recognition accuracy. In addition, in an LRRP SpeechWeb the natural-language query processors can be written in any language and can be executed on appropriately-sized computers at the remote sihlo sites. This architecture allows sophisticated speech applications, as well as simple "canned-answer" applications, to be easily added to a SpeechWeb with no need for end-users to have costly local devices. The following is an example of a SpeechWeb session:

AN EXAMPLE SESSION WITH AN LRRP SPEECHWEB

The user begins by accessing a sihlo whose address is input at the beginning of the session, or chosen through speech from bookmarked sihlos.

Suppose the initial sihlo is a bicycle sihlo. It responds by sending a recognition grammar to the browser on the user device and prompting the user:

bicycle sihlo: Hi, I am the bicycle expert.

User: What do you know about?
The user input is recognized using the bicycle grammar and sent to the bicycle sihlo.

bicycle Sihlo: Bicycle repair, mountain-bike trails and clubs.

User: Please tell me something about mountain-bike trails.
The browser is redirected to the mountain-bike-trail-sihlo whose grammar is downloaded

Mountain-bike-trail-sihlo: Hi, I know something about mountain-bike trails.
Where do you want to go cycling?.

User: Near Pontiac.
This user input is sent to the mountain-bike trail sihlo.

Mountain-bike-trail-sihlo: Sorry, input must be a Canadian Province or an American State

User: Michigan please.
The browser is redirected to the Michigan mountain-bike trail sihlo whose grammar is downloaded.

Sihlo: Hi, I know about some bike trails in Michigan.

User: Are there any trails near Pontiac?

Sihlo: Yes, there are four trails near Pontiac. etc.

A number of problems must be solved if LRRP SpeechWebs are to be widely employed as a means of providing speech access to distributed hyperlinked knowledge sources:

1. SpeechWeb browsers need to be easy to install and execute on conventional PCs, Laptops, cellphones, and other lightweight end-user devices.
2. Natural-language speech-recognition accuracy has to be improved for database-type queries in which little or no context is available to guide the recognizer.
3. Tools have to be developed to facilitate the construction of natural-language database query processors to help knowledge providers build sihlos.
4. Theories of natural-language semantics have to be extended to cover a wider range of database-query like expressions.
5. Protocols have to be developed to limit the way in which questions can be asked, and define a minimal set of questions that all sihlos must support (eg "what can I say").

In the following, we discuss these problems and present some partial solutions. We begin with a very practical issue and progress to more theoretical considerations.

2. Improving accessibility to LRRP SpeechWeb browsers

Over the last four years, we have experimented with a number of languages and communication protocols for building LRRP SpeechWeb browsers. Our initial browsers, which were demonstrated at PACLING [4], WECWIS [7], and AAAI [6 and 8], were built in Java using IBM's speech API's and communicated with remote sihlos through the Common Gateway Interface - CGI protocol. Our experience has led us to conclude that:

1. CGI is an appropriate protocol: it is supported by most operating systems, is easy to use, and requires no administrative layer, as does CORBA for example. Users can extend an LRRP SpeechWeb by placing natural-language processors and associated grammars in CGI directories in much

the same way as the conventional web is extended by adding html pages to web directories.

2. The SpeechWeb browser should not contain any proprietary components as we were unable to distribute our browser for use by others.

We decided to rebuild our browser as a single simple VXML script to be executed by a VXML interpreter running on the local end-user device. At first, this appeared to be straightforward. The browser was coded as two nested iterative loops: the outer loop being cycled whenever a redirect to a new sihlo is required; the inner loop being cycled for each user question. Recognition grammars were to be downloaded at the beginning of each cycle of the outer loop. Questions entered by the user were to be sent to the remote sihlos by use of a VXML command which invoked a Java object to send the question, via an HTTP Post command, over the Internet to the CGI host. Answers were to be returned from the host, through CGI, to the VXML browser which would output them through synthesized voice. Unfortunately, this approach did not work. The reason is that, in VXML, remote grammars are retrieved by commands that require the grammar location to be given as literal text. The address of the location cannot be changed at runtime. This problem cannot be overcome by using a fixed address of a local file which is used to hold the grammars and whose contents can be changed whenever a redirection to a new sihlo is issued to the browser. The reason is that VXML caches grammars when first retrieved. Therefore, even if the grammar in the local store is changed in each cycle through the outer loop, only the first grammar retrieved in a session will be used by the VXML recognizer.

Our solution to this practical problem was to design the LRRP browser so that, whenever a redirect to a new sihlo is required: 1) it calls a script that creates a copy of the browser containing a different grammar-retrieval command with the new location, and then 2) redirects the local VXML interpreter to use the new browser instead of itself. This solution enables the SpeechWeb browser to be constructed as a single VXML page with two simple associated Java objects. The approach has been successfully tested on laptops and PCs.

3. Improving accuracy through recognition-grammar design

Recognition accuracy is affected by both language size and branching factors. The size of a language is the number of expressions that can be generated by the defining grammar. The branching factor (perplexity) for a phrase is the number of different words with which that phrase can begin.

In addition to accuracy, another important property of a speech application is "robustness": the ability of the system to accommodate unexpected or grammatically-incorrect utterances. Robustness and accuracy are competing features with grammar-based recognition. It is reasonable to assume that: 1) the most-robust recognition grammar is a "word-sequence grammar" consisting of a single rule stating that the input consists of any sequence of words from the application vocabulary. The average perplexity is the size of the vocabulary, resulting in poor accuracy for anything other than simple applications. 2) The next most-robust grammar is a "syntactic grammar" containing rules that constrain the input language to utterances that are syntactically (grammatically) correct. Syntactic grammars provide significantly-greater accuracy than word-sequence grammars. 3) The least robust, but most-accurate grammar is a "semantic grammar" which constrains the input language to utterances that are semantically as well as syntactically correct; i.e. utterances that make sense to someone who knows the capability of the application.

What is not clear, is the level of accuracy that can be expected for different sizes of language, and the level of improvement in accuracy that can be achieved (at the expense of robustness) when adding syntactic and/or semantic constraints to a grammar. We were surprised to find that very little has been published on this topic despite the fact that commercial grammar-based speech-recognition software has been readily-available for over five years. (Numerous useful tutorial-style papers on voice application design are available on the web (e.g. IBM [12]), but they say little about this aspect of grammar design. We did discover some research that was carried out using different recognition technology, which showed that users have reasonably high tolerance for reduced vocabularies [15]. However, that result is only relevant for a limited type of application.

We decided, therefore, to undertake an experimental investigation of grammar design using our LRRP SpeechWeb browser. We used three grammars in one experiment, and then extended them by increasing their vocabularies in a second experiment. The database contained some geographic information as well as information about the solar system. Example queries defined by the semantic grammar are: "How many moons orbit mars?" "Which moons were discovered by Kuiper or Hall?", "Does every moon orbit a red planet?", "Which city is the capital of China?"

Grammar	Language size	Branching factor
semantic	$2.7 * 10^{12}$	39.6
syntactic	$3.05* 10^{15}$	95.5
word sequence	$2.31* 10^{24}$	273
extended grammars		
semantic	$5.55* 10^{12}$	95.6
syntactic	$8.17* 10^{15}$	267.3
word sequence	$2.40* 10^{27}$	547

Two subjects took part in the experiments: a male native-English speaker and a female non-native-English speaker. Three types of database queries were used as sample utterances: 1) queries that were semantically (i.e. made sense) as well as syntactically correct, 2) queries that were only syntactically correct, and 3) queries that consisted of sequences of words from the vocabulary that were neither semantically nor syntactically correct. The complete results of the investigation are available in a Master's thesis [20]. The following table summarizes some of the results for the extended grammars:

Type of queries	Grammar used	Correct	Incorrect	Not recognized
semantically correct	semantic	75%	4	21
	syntactic	66	14	20
	word sequence	12	60	28
syntactically correct	semantic	0	22	78
	syntactic	65	5	30
	word sequence	8	44	48
neither	semantic	0	10	90
	syntactic	0	30	70
	word sequence	15	56	29

Many of the results were as expected. The semantic grammar was the most accurate for semantically-correct queries, and the least robust for the other types of query. The syntactic grammar was the most accurate for syntactically-correct utterances. The word-sequence grammar was the least accurate for all but the word-sequence queries, where it was able to correctly recognize a small percentage of the utterances. There were three unexpected results: 1) the accuracy of the semantic and syntactic grammars were both unexpectedly high (for semantically and syntactically-correct queries respectively) given the perplexity and size of the languages defined by these grammars, 2) the improvement in accuracy of the semantic grammar compared to the syntactic grammar was largely a result of a significant decrease in misrecognized queries rather than a decrease in not-recognized queries (this is important as it is easier to deal with not-recognized rather than misrecognized utterances), and 3) the word-sequence grammar was able to correctly "spot" a reasonable number of words in the input despite defining a very large language.

4. Constructing natural-language database-query processors as executable specifications of attribute grammars.

Although simple sihlos can be easily constructed in any programming language, there is a need for software to facilitate the construction of sophisticated natural-language query processors. One approach is to transform the queries into SQL or some other formal database-query language. However, this introduces a level of indirection, reduces the modularity of the processor, and limits the range of queries that can be accommodated owing SQL's inability to handle general negation, modality, and intensionality. Other approaches that are based on keyword and phrase-matching techniques are of no use for questions that require answers to be computed from the contents of one or more data stores. Such computation has to be "compositional" in the sense that the answer to the whole query must be computed from the meanings of its parts, using some well-defined rules.

When a large variety of query structures have to be accommodated, the rules used to compute meanings are usually applied according to the syntactic structure of the query. (The alternative would be to have an unmanageably-large set of query patterns.) Such syntax-directed evaluation is regularly used in the processing of programming and command languages. Building a program to parse the input and apply semantic rules to compute an answer is not an easy task. In order to facilitate this process, such programs can be constructed as executable specifications of attribute grammars which define the syntax and the semantics of the query language [17]. Using this approach, parsing and semantic-rule application is carried out by library functions that are hidden from the programmer. In or-

der to test the viability of this approach in the context of an LRRP SpeechWeb, we used the Windsor Attribute Grammar Programming Environment W/AGE [5]. This environment can be used by people with relatively-little programming experience to construct advanced natural-language query interfaces.

The following "attributed rule" is part of a W/AGE program that computes the answer to simple queries such as "who discovered a moon that orbits a red planet?" The rule states that a simple noun clause consists of a common noun orelse a number of adjectives followed by a common noun, in which case the value returned is obtained by intersecting the value of the adjectives and the value of the common noun:

```
snouncla = cnoun
           $orelse structure(s1 adjs ++ s2 cnoun)
           [a_rule (NOUNCLA_VAL $of lhs EQ intersect[ADJ_VAL $of s1, NOUNCLA_VAL $of s2]]

Example application:
snouncla (tokenize "red planet spins") =>[[[NOUNCLA_VAL [e_mars]], [WORD "spins"]]]
```

The complete program is approximately 800 lines long and can answer over 120,000 questions. We have used W/AGE to build other natural-language processors that we have embedded in sihlos and added to the experimental SpeechWeb. Owing to the fact that W/AGE programs are interpreted, it was necessary to write simple Unix scripts to allow the W/AGE processors to be invoked through the CGI protocol on the host machines where the sihlos were stored.

5. Improving expressiveness: a compositional semantics for natural-language queries

No matter how the natural-language query processors are built, we must first begin with a compositional semantics for the subset of natural-language to be used.

5.1 Difficulty of developing a compositional semantics

The development of such semantics is not easy. For example, consider the queries "does phobos spin?" and "does every planet spin?" A naive approach would be to have intransitive verbs and common nouns, such as "spin" and "planet" denote unary relations; and proper nouns such as "phobos" denote entities. The rules of composition would then be as follows, where `||x||` stands for the denotation of `x`.

```
query ::= "does" proper_noun intransitive_verb
answer = True
          if ||proper_noun||
          is_a_member_of ||intransitive_verb||
         = False otherwise
```

```
query::= "does" "every" common_noun
                                intransitive_verb
answer = True
          if ||common_noun||
          is_a_subset_of ||intransitive_verb||
         = False otherwise
```

Now consider extending these rules to accommodate queries such as "does phobos and every planet spin?", "does phobos and deimos spin?" etc. Do we need to define a different rule for each of these types of query? If so, we will need hundreds of rules for even a relatively small query language. The solution is to identify a "small" grammar that covers the query language, and a matching "semantic theory" which assigns a single semantic rule to each of the syntax rules in the grammar. This difficult task is further compounded by other features of natural language including ambiguity, modality, and intensionality.

5.2 Use of Montague semantics

We have investigated various semantic theories as a basis for the sihlo natural-language query processors. Our investigation has led us to recommend the compositional semantics of Richard Montague [16 and 2]. Montague's semantics is based on the concept that words and phrases denote functions in a function space constructed over the boolean values `True` and `False`, an infinite set of entities, and a set of possible worlds. No word, or pfrase, denotes an entity directly. As a simple example, proper nouns such as "phobos" denote higher-order functions that take properties such as that denoted by "spins" as argument and return `True` if that property is true of the entity `e_phobos` (which is represented internally). Determiners such as "every" denote higher-order functions which take two properties `p1` and

p2 as arguments and return a boolean value. For example, ||every||returns True if every entity that has property p1 also has property p2. This approach solves the problem of combining denotations of termphrases discussed above, owing to the fact that ||phobos|| and ||every moon|| are now functions of the same type (both take a property as argument and return a Boolean). The denotation of the word "and" can now be defined as follows:

```
and f g = h where h p = (f p) & (g p)
```

That is, ||and||is a higher-order function that takes two functions fand g (e.g. denoted by "phobos" and "every moon") and returns a function h such that the result of applying h to a property p (e.g denoted by "spins") is obtained by applying f to p, and g to p, and then conjoining (&) their results.

Montague's semantics provides a highly orthogonal compositional semantics for many aspects of natural language including quantification, and modal and intensional expressions.

5.3 Implementing Montague semantics in the natural-language query processors

Montague was not concerned with implementation and his semantics cannot be used directly in a natural-language interface as it is computationally intractable. However, a tractable subset of Montague semantics can be readily implemented by replacing characteristic functions of sets by the sets themselves, and modifying all other denotations accordingly.

In our investigation, we coded a tractable version of Montague semantics in the attribute rules of an executable specification of an attribute grammar written in W/AGE. The coding was straightforward for two reasons: 1) Montague's approach, of associating semantic rules with syntactic rules of English, is very similar in concept to that of an attribute grammar, and 2) W/AGE is implemented as a set of library functions that are added to a higher-order functional programming language. Montague's functional denotations can easily be coded directly in that functional language.

Montague did not provide complete details for the denotations of some natural-language constructs and therefore we had to fill in some parts. For example, we developed a compositional semantics for negation [3] using ideas from Montague and Iwanska [13]. We also developed two compositional semantic schemes for adjective-noun combinations [1]. In both of these endeavors, we found that the use of a higher-order functional-programming language greatly simplified our task.

6. Conclusions and future work

We have described a new (LRRP) SpeechWeb architecture. We have explained how it differs from other architectures currently in use, and why it is preferable especially for use with lightweight end-user devices. Our investigation has also demonstrated that:

1. Conventional web services, and the simple CGI protocol, are sufficient to support an LRRP SpeechWeb that supports simple queries.
2. An LRRP SpeechWeb browser can be constructed as a single VXML page executing on an end-user device. Application-specific recognition grammars can be downloaded in realtime and used to recognize speech locally.
3. Commercial grammar-based speech recognizers, such as that used in IBM's VXML interpreter, can achieve good accuracy for real applications with query languages having billions of queries.
4. Recognition accuracy can be improved by encoding semantic constraints in the syntax rules of grammar-based speech recognition.
5. Responsive natural-language query processors are relatively easy to construct as executable specifications of attribute grammars.
6. Montague Semantics is useful as the basis for natural-language query processing.

Currently, we are:

1. Developing a site from which our browser and the W/AGE environment can be freely downloaded.
2. Extending the compositional semantics to accommodate more complex verb phrases [19].
3. Investigating ways to generate voice data-input applications automatically from XML schemas [9].
4. Investigating weighted combined grammars in LRRP SpeechWeb browsers. This may lead to good accuracy together with good robustness.
5. Investigating other communication protocols in order to support dialogues.

As soon as J2ME/VXML-enabled cellphones become available, we will port our LRRP SpeechWeb browser onto these phones and provide the first-ever flexible, scalable and economic SpeechWeb that allows speech services to be deployed as easily as conventional web services. The major challenge will then be how to encourage widespread use of the technology, in order to facilitate the creation of a large public-domain Speechweb.

Acknowledgements

The authors acknowledge the support provided by the Natuaral Sciences and Engineering Council of Canada.

References

[1] Abdullah, N., "Two compositional semantics for adjective/noun combinations", *Master's Thesis*, School of Computer Science, University of Windsor, ON, Canada, 2003.

[2] Dowty, D. R., Wall, R. E. and Peters, S. *Introduction to Montague Semantics*. D. Reidel Publishing Company, Dordrecht, Boston, Lancaster, Tokyo, 1983.

[3] Frost, R. A., and Boulos, P., "An efficient compositional semantics for natural-language database queries with arbitrarily-nested quantification and negation", *Proceedings of the 15th Canadian Conference on Artificial Intelligence, AI'2002*, May 2002, University of Calgary, Alberta. Lecture Notes in Artificial Intelligence, No. LNAI 2338. Eds. Cohen and Spencer. Springer-Verlag, Berlin, 2002, pp. 252–267.

[4] Frost, R. A.. and Chitte, S., "A New Approach for Providing Natural-Language Speech Access to Large Knowledge Bases", *Proc. of PACLING '99, The Conference of the Pacific Association for Computational Linguistics*, 1999, pp. 82–90.

[5] Frost, R. A., "W/AGE The Windsor Attribute Grammar Programming Environment", *IEEE Symposia on Human Centric Computing Languages and Environments HCC'2002*, 2002, pp. 96–98.

[6] Frost, R. A., "SpeechWeb: A web of natural-language speech applications", *Proceedings of the AAAI '02 Intelligent Systems Demonstrations (ISD)*, U. of Alberta, Edmonton, 2002, pp. 998–999.

[7] Frost, R. A., "SpeechNet: A network of hyperlinked speech-accessible objects", *Proceedings of the IEEE WECWIS International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems. Joint Workshop of (3rd RTDB and 2nd DARE)*, San Jose, April 1999, pp. 71–76.

[8] Frost, R. A., "A Natural-Language Speech Interface Constructed Entirely as a Set of Executable Specifications", In *Proceedings of the Intelligent Systems Demonstrations, of the Eleventh National Conference on Artificial Intelligence, 908–909. AAAI'99* , Orlando, Florida: AAAI Press, 1999, pp. 908–909.

[9] Hanna, F., "Survey of automatic generation of voice applications", School of Computer Science, *60–510 Survey Report.* University of Windsor, 2003.

[10] Hartzell, D., "Overcoming hurdles for deployment of speech services", Wireless Future Magazine. Janury/February 2003 issue.

[11] Hartzell, D., "Invited Talk: Simplyfying speech-enabled solutions — or deploying speech-enabled services should be as easy as deploying web services", *Voice Enabled Services 2003*. London U.K. 13th-15th Jan. 2003.

[12] IBM http://www-106.ibm.com/developerworks/library/i-voicestudio/ 2003.

[13] Iwanska, L., "A General Semantic Model of Negation in Natural Language: Representation and Inference", *Doctoral Thesis*, Computer Science, University of Illinois at Urbana-Champaign, 1992.

[14] Lucas, B.,"VoiceXML for web-based distributed conversational applications", *Communications of the ACM* 43 (9), 2000, pp. 53–57.

[15] Moody, T. S., "The effects of restricted vocabulary size on voice discourse structures", *Ph.D. Thesis*, North Carolina State University, 1998.

[16] Montague, R., *Formal Philosophy: Selected Papers of Richard Montague*, edited by R. H. Thomason. Yale University Press, New Haven CT, 1974.

[17] Paaki, J., "Attribute Grammar Paradigms — a High-Level Methodology in Language Implementation", *ACM Computing Surveys*, 27 (2), 1995, pp. 196–256.

[18] PipeBeach : www.pipebeach.com 2003

[19] Roy, M. *Master's Thesis Proposal* School of Computer Science, University of Windsor, ON, Canada, 2003.

[20] Shi, Y., "Investigation of Recognition-Grammar Design" *Master's Thesis*, School of Computer Science, University of Windsor, ON, Canada, 2003.