Associating the Dialogue Move Engine with Speech Input

Jim Hieronymus Staffan Larsson Leif Grönqvist

Distribution: Public

Specification, Interaction and Reconfiguration in
Dialogue Understanding Systems: IST-1999-10516

Deliverable D2.1

December 2000

Specification, Interaction and Reconfiguration in Dialogue Understanding Systems:
IST-1999-10516

Göteborg University
Department of Linguistics

SRI Cambridge
Natural Language Processing Group

Telefónica Investigación y Desarrollo SA Unipersonal
Speech Technology Division

Universität des Saarlandes
Department of Computational Linguistics

Universidad de Sevilla
Julietta Research Group in Natural Language Processing

For copies of reports, updates on project activities and other SIRIDUS-related information, contact:

The SIRIDUS Project Administrator
SRI International
23 Millers Yard,
Mill Lane,
Cambridge, United Kingdom
CB2 1RQ
milward@cam.sri.com

See also our internet homepage http://www.cam.sri.com/siridus

Contents

Chapter 1

Introduction

The purpose of this deliverable is to describe the connection of system speech input to information states in the baseline SIRIDUS architecture. This important step will allow us to explore ways in which information states can help guide and refine the speech recognition.

We hope that using knowledge of the previous system dialogue move to change the probility of the words and grammars to bias them toward the expected answers, will increase the system accuracy, robustness and speed. Since the exact response will never be completely predictable, it may be best to run several recognizers in parallel, and choose the one with the best match scores, or the most plausible parses.

The ideal system would be one in which the system asks for some information, and the system has increased sensitivity to the expected answers, while retaining recognition suited to clarification questions and answers to other related questions. Just where and how to optimally implement these capabilities within the dialogue system is the subject of this research.

Chapter 2

Using information states to improve recognition

Connecting Information States with speech input allows the Siridus project to explore the usefulness of coordinating dialogue moves and information state updates with the speech recognizer. In future lattice-based systems, the speech recognizer provides the dialogue move engine with word lattices (presently derived from N-best lists), which have the possibility of having multiple confusable words occupying each interval in time. Because the speech recognizer has a language model loosely related to the task domain, these alternatives are plausible. The syntax and semantic component can then analyze the word lattice to find the most probable "correct sentence" given the previous context and the present state of the dialogue. This should have the effect of making the system more robust to the great diversity of human language even within a limited task domain, to relaxed styles of speaking which reduce the distinctiveness of words, and to noisy environments which makes the speech recognition less accurate overall. All three of these effects are known to adversely effect speech recognition performance in for example the "How may I help you" system at AT&T Research [6], the Switchboard Speech Recognition evaluations [13] and the speech recognition in noise studies at Cambridge University [5] respectively.

Chapter 3

Selection of interfaces to the DME and Information State Update System

The present SIRIDUS baseline system is written in Prolog. Therefore it is best to find an interface which is capable of connecting to the present system and to components which may be written in C and C++ (the speech recognizer and synthesizer) and in JAVA (the user interface). Since the dialogue system can be viewed as a set of agents, it would be interesting to use an already developed agent system. We have looked at Open Agent Architecture (OAA) which was developed at SRI and was discussed in D6.1. At Gothenburg University a simpler system called AE was developed which was designed to be fast and simple. In initial tests we found that OAA was considerably slower than AE for updating information states, which involves moving rather large data structures. This problem with moving large amounts of data may be a bug in OAA and is being examined by the OAA developers. So we think it is best that the initial system will consist of a subset of AE components (dialogue move engine and dialogue manager) which together transfer large data structures between them and need the extra efficiency of AE. This unit together will act as an OAA agent. Thus high data transfer paths would be controlled by AE, in this implementation. The initial implementation of the Speech Recognition Interface to the DME system will use OAA.

The task is then to provide an OAA wrapper for the speech recognition system, which allows the speech recognizer to be controlled by the Siridus dialogue manager, and writes the word lattice to the parser agent for subsequent processing. The OAA system can run on Linux or Solaris operating systems. The initial agents are

- speech recognizer
- parser

- semantic interpreter
- dialogue manager
- generator
- speech synthesizer

Systems will also include other components, for example, a database query component or perhaps a component capable of executing commands.

Chapter 4

Choosing the speech recognizer

There are a number of speech recognizers which could be used for the Siridus system, each with particular advantages and disadvantages. The recognizers considered seriously are Decipher, Nuance, CMU Sphinx II and III, IBM, and HTK. We had a design goal of having the whole system run under Linux, so this was one of the major considerations. So far the possible systems offer a matrix of capabilites and shortcomings. In tables 4.1 and 4.2 are matrixes of the systems we considered,

Recognizer	Linux	Solaris	Windows/NT
Decipher		X	
Nuance		X	X
CMU Sphinx	X	X	X
HTK	X	X	X
IBM	X		X
L & H			X
Microsoft			X

Table 4.1: Platform availability

Recognizer	Spanish	English	Swedish
Decipher		X	(X)
Nuance	X	X	X
CMU Sphinx		X	
HTK		X	
IBM	X	X	
L & H	X	X	X
Microsoft		X	

Table 4.2: Language coverage

Recognizer	output type	grammar type	barge-in support	speaker independent
Decipher	(lattice)	FS, ngram unification		X
Nuance	n-best string	FS, n-gram, unification	X	X
CMU Sphinx	lattice,nbest	n-gram	X	X
HTK	lattice,nbest	FS,n-gram	X	X
IBM	single string	FS		X
L & H	string	FS		X
Microsoft	string	FS		X

Table 4.3: Miscellaneous properties (FS=Finite State)

The Decipher recognizer comes from many years of research at the STAR lab at SRI USA. It has undergone many years of development after the spin off of Nuance from SRI in 1994. The recognizer is a continuous density mixture Gaussian Hidden Markov Model system which uses vector quantization for speeding up the Euclidean distance calculation for probability estimation. The system uses context dependent triphonic cross word acoustic models with speaker normalization based on vocal tract length normalization, channel adaptation using mean Cepstral subtraction and speaker adaptation using Maximum Liklihood Linear Regression (MLLR) (first developed at Cambridge University [12]). There are approximately 25,000 triphonic (context conditioned phonemes) models in the English system. (This description fits most of the high performance speech recognition systems available today.) It is available to the Siridus project because SRI Cambridge is a partner. This system is capable of producing word lattices, but it cannot do so live input real time mode. Word lattices are only available in batch mode recognition. This recognizer has both American English and Swedish acoustic models, but thus far only the English models are available to us. Decipher has been described in various publications [14] and [18]. This system has been in the Switchboard competitions and has incorporated various linguistic techniques to improve robustness, including disfluency detection and correction, and topic change detection [17] This recognizer is available to the Siridus Project for Sun Solaris.

The Nuance system [19] was developed from the Decipher system, beginning in 1995. Since then it has evolved in its own way, with emphasis on real time performance, robust speech recognition over telephones (land lines and mobile). Most of the commercial systems which they have produced have a finite state grammar structure which is interesting for advanced dialogue systems. The finite state grammar involves collecting a large amount of field test data, and then writing grammare rules to cover all of the sentence structures seen in the task domain. The difficulty is that humans have a rich set of possible ways of expressing the same idea. In a very limited task, this can be covered with a finite state grammar, with the side effect that the grammar is very fragile. The Nuance system can also be used to recognize speech using an ngram word sequence grammar or a compiled Gemini Unification grammar. The latter can be a very general purpose grammar of spoken English which can then be specialized to a particular task domain. Recogni-

tion modules for several languages have been developed and are currently available these include English (US, British, Canadian, Australian, New Zealand, Singapore and South African), French (Canadian, Coninental) , German, Greek, Italian, Japanese, Brazilian Portuguese, Spanish (European and Latin American), and Swedish. These acoustic models are available to the Siridus project under a developers license. The recognizer is available for Windows NT and Sun Solaris. In order to do echo cancellation and barge in it is necessary to buy the very expensive Dialogics Antares telephone interface cards.

The CMU system has two different versions, Sphinx II and Sphinx III available as open source. The Sphinx II system is a semi-continuous Hidden Markov Model which uses mixtures of vector quantizers to do the probability calculations for triphonic acoustic models. [9] The vector quantized system is fast but not as accurate as the Gaussian mixture system. Sphinx III system is a Gaussian mixture system. This is in the process of being rewritten for real time performance. The language models which are available are for American English only. The system is available for telephone dialogues with full duplex audio and hardware echo cancellation, which is necessary for barge in. The system runs under NT and may be available for Linux. The system is capable of outputing word lattices. CMU has put most effort into running their system under NT because there are telephone interface boards which do hardware echo cancellation which run under NT. The current telephone board is Linejack from Quicknet Technologies. This system has been used in their DARPA Communicator sytstem because it allows the cancellation of the sidetone from the speech output from the system. The Linejack card is now available with Linux drivers.

The Cambridge University Hidden Markov Model Tool Kit based speech recognizer has been under development for many years. The recognition system typically comes out best in the DARPA funded performance tests, for various speech recognition domains including Broadcast News, Telephone Spontaneous speech, Wall Street Journal Dictation, etc. The basic tool kit without acoustic models is available in the public domain. However the best “Odell decoder” is not generally available [16]. Microsoft has bought Entropic which made the HTK system commercially available previously. The HTK system can produce word lattices, and some high quality acoustic models are available in the HAPI system which was available from Entropic. The recognizer runs on Linux, NT, and Sun Solaris. The available system does not allow multiple pass word lattice decoding which generally produces the best results. The HTK system incorporates speaker and channel adaptation using MLLR as mentioned earlier. There is now an ongoing effort to make more of the CU system available in the public domain, hopefully including the “Odell decoder”.

IBM has conducted research on automatic speech recogntion for more than 50 years. The Hidden Markov Model technique was pioneered by Jelenek and the speech processing group at the IBM Watson Lab ([10],[2]) and many of the techniques used today originated from this group. The present IBM system ViaVoice recognizer is a Gaussian mixture HMM system based on context dependent sub-phone units, which can be shared amoung tri-phone units.([4]) The system is initially speaker independent, but can adapt continuously to the user’s voice. IBM offers ViaVoice dictation systems in many languages, including English (UK and US), Mandarin Chinese,

French, German, Italian, Japanese, and Spanish. ViaVoice has a Linux version which is available to researchers for non-commercial use. There are developer toolkits available for finite state grammar construction.

Learnout and Hauspie [11] has produced a speech recognizer which is available for English, French, Dutch, German and Swedish. Recently L&H bought Dragon Systems which is famous for its excellent dictation systems. It is not clear if the present L&H recognizer is based on the Dragon system. While the acoustic models are available, task language models are not available. The recognizer for each language costs $ 1,000 for a single user license. The recognizer runs finite state grammars, and does not seem capable of ngram word sequence grammars. The capability and language modeling options for this system is presently under investigation by Gothenburg University. Recently L&H has declared Bankrupcy, which clouds the future availability and usefulness of their systems.

Microsoft is incorporating a speech recognition system in their next generation operating system, Windows 2000 plus, code named Whisler. While the group at Microsoft has published some papers on its work, it is not clear what the details of their curent recognition system are. The system is derived from Sphinx II-III, which was seveloped by the CMU Speech Recognition Group which then left to join Microsoft Research. [7] [8] The first system was a semi-continuous Hidden Markov Model system, but the present system is a Gaussian mixture continuous density HMM system with cross word triphonic models. Much work has gone into making it fast, and to use very little memory. The latter effort seems to have been mostly misguided by an outdated view of the cost and availability of memory on PC's. What languague models will be available is largely unknown, as is the exact applications which Microsoft is targeting. It seems unlikely that word lattices are available, since they removed word lattices from their latest speech applications interface (SAPI). This system will not run on Linux.

Because the multilanguage capability is important to an European Union project we have decided to make the first system a Nuance recognizer. This will initially run in a mixed OS environment, with each component perhaps running on a different OS. The OAA architecture will allow this configuration to be used with each component running in its best environemnt. It should be easy for Nuance to do a Linux port of their system, but at present they have not made a commitment to support this system. The Nuance system can also compile Gemini Unification grammars as recognition grammars. This will allow a direct comparision of ngram grammars and Gemini grammars for our dialogue systems. An issue here is whether it is more robust to have a more aggressive grammar in the recognizer and then do the analysis, or whether it is better to have an n-gram word grammar which recognizes more malformed utterances, and then do the analysis. The other feature of the present Nuance system is that it does not provide word lattices. We will use its nbest capability to manufacture word lattices at first, perhaps in the future Nuance will offer word lattices as an output option.

At a later date it will be possible to provide OAA wrappers for other promising speech recognizers, with the second system likely to be the HTK system and then perhaps the Sphinx system.

This would allow direct tests of various systems to find the one which performed best for our use. The HTK system also allows word lattices, which is a priority for us. Unfortunately the best HTK system, the Odell decoder, is not yet available publically.

Chapter 5

Language Models

The travel information task domain requires a language model for the speech recognizer. At present there is very little speech data on natural travel information dialogues to train such a model. Thus is is necessary to use other data from spontaneous corpora to provide enough data to train an n-gram language model. Another way to make the best use of the small amount of task related data, is to make a category n-gram model with categories related to the task domain. For the travel information domain, so that the categories can be populated with more category items. Because the airports and destinations are different in the European Union, categories like CityName, AirportName, AirlineCompany, etc. can be augmented with names more appropriate to Europe. This allows us to use the data we have to make a language model appropriate for the location of our task domain. This was done using the CMU-Cambridge language modeling tool kit, [3] and resulted in perplexities of around 16 when tested on a jackknifed training and test set. Of course much of the branch perplexities are hidden in the categories by using this technique, with the branching for CityNames actually containing 100 names at present. This is a very good way of using limited training data to construct a broad coverage language model. We have decided to use the CMU Human-Human Dialogues which were collected for the Communicator project and the OGI stories corpus of spontaneous monologues to train the language model. [15]

Chapter 6

Vocabulary for Recognition

The vocabulary for the recognizer consists of all of the words found in the training data, plus all of the words which have been added in the categories for the category grammar. Using the words which were used by the travel agents is important, because the users are likely to echo part of the travel agent's utterances, and use similar constructions and expressions. Systems which do not use this expanded grammar are in danger of failing when the refering expression used by the travel agent is absent from the vocabulary. As the categories are expanded one has to be careful about adding them to the vocabulary. There are a collection of categories depending on the exact task. Here CityName, AirportName, AirlineName, etc. need to populated with the appropriate items for Europe. As the task expands, more CityNames may be added.

Chapter 7

Changing the Language Model

In order to alter the language model according to the information state, the language model will have to be changed on the fly or run in different recognizers. We favor changing the ngram language model on the fly while leaving the acoustic models and the vocabulary the same. This just makes the probability of certain things, all of the possible answers to what city do you want to fly for example to have a higher probability. In fact by changing the unigram probability, one can in effect change the vocabulary.

Speed may be an issue. This is the reason that most of the systems prefer to use a finite state network. We would like to use word lattices as the tranfer data structure to the parsers, and have two parsers running, a complete robust parser and a pieces parser. There would be some sort of time limit and some heuristics to patch the pieces together into something sensible.

Chapter 8

Baseline speech input module

Here, we describe the setup enabling speech input for the baseline demo system. It runs under Linux and uses the IBM ViaVoice ASR system and works with the terminal based TrindiKit.

There are two basic ways to interface the recogniser to the SIRIDUS baseline architecture. Either one builds an OAA wrapper and runs the recogniser as an OAA agent, or one builds a TrindiKit wrapper and runs the recogniser as a TrindiKit module. For the baseline implementation, we have done the latter.

For speech input, a TrindiKit module called `input_viaasr` has been built. The ViaVoice recognizer will run as a background process started from a C program. The `input_viaasr` module has an `init` predicate that starts the recognizer process (executes the C program) and asserts the in- and out-pipes to the prolog database, a `quit` predicate that cleans up and halts the recognizer (sends '`q`' to the C program), and a `listen` predicate that reads the current recognized utterance. The `quit` predicate has to be called from `start.pl` in the dialogue system and the new input module will replace the text input module.

To be able to test the ViaVoice system we have made a simple test grammar (see Figure 8.1). The grammar is compiled by a ViaVoice tool using the call `vtbnfc -m+ -o travel.fsg travel.bnf`).

The C program is prepared to be able to load a huge dictation grammar instead of the simple travel agency grammar but it does not currently perform very well. This may depend on the recogniser but may also be due to problems with the soundcard, the soundcard driver, or the microphone setup. The soundcard has to be able to use full duplex. Otherwise speech input and speech output will not work at the same time.

```
<sample> = <greet>? <travelby>? <prepcity>? <prepcity>?
           ((<inmonth>? <class>?) | (<class>? <inmonth>?) <extra>?) |
           <answer> <extra>? .

<prepcity> = <prep>? <city> .
<greet> = hello|hi|bye|goodbye .
<answer> = yes|no .
<class> = ((economy|first|second) class?) | (as? (cheap|expensive)
          (as possible)?) .
<extra> = please|thanks .
<prep> = to|from .
<travelby> = flight|bus|train .
<city> = paris|london|stockholm|boston|washington .
<month> = january|february|march|april|may|june|july|august|september|
          october|november| december .
<inmonth> = in? <month> .
```

Figure 8.1: ViaVoice test grammar

Bibliography

[1] American Association for Artificial Intelligence. *Proceedings of the Fifth National Conference on Artificial Intelligence*, Philadelphia, Pennsylvania, August 1986.

[2] L. R. Bahl, F. Jelinek, and R. L. Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):179–190, 1983.

[3] P.R. Clarkson and R. Rosenfeld. Statistical language modeling using the cmu-cambridge toolki. In *Proc. Eurospeech97*, 1997.

[4] E. Eide, B. Maison, M. Gales, R. Gopinath, S. Chen, P. Olsen, D. Kanevsky, M. Novak, and L. Mangu. The ibm's 10x real-time broadcast news transcription transcription system used in the 1999 hub4 evaluation. In *Proc. 2000 Speech Transcription Workshop*, College Park, MD, 2000. Available at `http://www.nist.gov/speech/publications/tw00/index.html`.

[5] J. A. N. Flores and S. Young. Continuous speech recognition in noise using spectral subtraction and hmm adaptation. In *Proc. of ICASSP*, pages 409–413. IEEE, 1993.

[6] A. L. Gorin, G. Riccardi, and J. H. Wright. How may i help you? *Speech Communication*, 23:113–127, October 1997.

[7] X. Huang, A. Acero, F. Alleva, D. Beeferman, M. Hwang, and M. Mahajan. From cmu sphinx-ii to microsoft whisper: Making speech recognition usable. Tech Report MSR-TR-94-20, Microsoft Research, Redmond, WA, September 1994.

[8] X. Huang, A. Acero, F. Alleva, M. Y. Hwang, L. Jiang, and Mahajan M. Microsoft windows highly intelligent speech recognizer: Whisper. In *Proc. of ICASSP95*, May 1995.

[9] X. D. Huang. *Semi-continuous Hidden Markov Modeling for Large Vocabulary Speech Recognition*. PhD thesis, University of Edinburgh, 1992.

[10] F. Jelinek. Continuous speech recognition by statistical methods. *IEEE Proceedings*, 64(4):532–556, 1976.

[11] Learnout and Hauspie (www.lhs.com). Website. Technical report, Learnout and Hauspie, Ieper, Belgium, 2001.

[12] C.J. Leggetter and P.C. Woodland. A representation for collections of temporal intervals. In AAAI-86 [1], pages 367–371.

[13] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The det curve in assessment of detection task performance. In *Proceedings of Eurospeech-97*, pages 1895–1898, 1997.

[14] H. Murveit, J. Butzberger, V. Digalakis, and M. M. Weintraub. Large-vocabulary dictation using sri's decipher(tm) speech recognition system: Progressive-search techniques. In *Proc. of ICASSP93*, volume II, pages 319–322. IEEE, 1993.

[15] Y.K. Muthusamy, R.A. Cole, and B.T. Oshika. The ogi multi-language telephone speech corpus. In *Proc. of ICSLP92*, volume 2, pages 895–898, 1992.

[16] J. Odell. *THE USE OF CONTEXT IN LARGE VOCABULARY SPEECH RECOGNITION*. PhD thesis, University Cambridge, 1995.

[17] E. Shriberg, R. Bates, and A. Stolcke. A prosody-only decision-tree model for disfluency detection. In *Proc. EUROSPEECH 97*, volume 5, pages 2383–2386, 1997.

[18] A. Stolcke, H. Bratt, J. Butzberger, H. Franco, V. R. Rao Gadde, M. Plauche, C. Richey, E. Shriberg, K. Sonmez, F. Weng, and Zheng J. The sri march 2000 hub-5 conversational speech transcription system. In *Proc. NIST Speech Transcription Workshop, College Park, MD*, 2000.

[19] Nuance Communciations (www.nuance.com). Developer's manual. User Manual Version 7.0, Nuance Communciations, Menlo Park, CA, 2000.