Hand-out for the COST249 meeting in Rome, 17-18 febr. 1997

LOCALIZING A SPOKEN DIALOG SYSTEM FOR TRAIN TIMETABLE INFORMATION

Helmer Strik, Albert Russel, Henk van den Heuvel, Catia Cucchiarini, and Lou Boves

Department of Language and Speech, University of Nijmegen, The Netherlands

ABSTRACT

This paper reports on the development of a spoken dialog system for providing information about public transport in the Netherlands. It is explained how a German prototype was adapted for Dutch. Emphasis is laid on the specific approach chosen to collect speech material that could be used to gradually improve the system. The pros and cons of this method are discussed.

1. INTRODUCTION

During the last decade the performance of spoken dialog systems has improved substantially. At the moment, the quality of these systems seems to be able to support a number of simple practical tasks in small and clearly delimited domains. As a result, much effort is spent nowadays to develop prototype telephone-based information systems in different countries. These systems are reminiscent of the well-known Air Travel Information System (ATIS) task that has been a focal point in the American ARPA-project. In Europe two MLAP (Multi-Lingual Action Plan) projects concerning public railway information have been carried out, viz. RAILTEL and MAIS. These projects differ from the ATIS task in that they aim to construct truly interactive systems, accessible over the telephone.

There are many reasons why information about public transport is a suitable domain for testing spoken dialog systems, of which only some are mentioned here. First of all, the domain can be limited in ways that are obvious for a caller, which is a necessary requirement to reach a sufficient performance level. In the Dutch system that we are developing, the domain is limited by restricting the information to travels between train stations. Furthermore, there is a large demand for information about public transport. For instance, in the Netherlands there is one nationwide telephone number for information about public transport. This number receives about 12 million calls a year, of which only about 9 million are actually answered. At the moment, all calls are handled by human operators. A substantial cost saving would be achieved if part of these calls could be handled automatically. Moreover, automatic handling would probably reduce the number of unsuccessful calls.

In the Netherlands ‘public transport information’ is virtually identical with ‘multi-modal from address-to-address information’. The human operators who provide the service must access a large data base that contains the schedule information of all public transport companies in the country. Especially the fine-meshed local transport networks pose substantial problems, e.g. when specific bus stops must be identified. The complex dialogs that may be required to disambiguate destinations on the address level are far beyond what can be achieved with existing speech recognition, natural language processing, and dialog management technology. Therefore, we have limited the domain of our experimental public transport information system to information about travels between train stations. However, we intend to enlarge that domain gradually, e.g. by adding metro stations in Amsterdam and Rotterdam, and by adding tram stops and the major inter-regional buses (Interliners).

2. GENERAL DESCRIPTION OF THE SDS

The starting point of our research was a prototype developed by Philips Research Labs (Aachen, Germany), which can provide information about the schedules of the German railways. In this section only a short description of the general properties of the system is given. For further details concerning this system, see Oerder and Ney (1993), Steinbiss et al. (1993), Aust et al. (1994), Ney and Aubert (1994), Oerder and Aust (1994), Steinbiss et al. (1994), Aust et al. (1995), Steinbiss et al. (1995), and Strik et al. (1996). Conceptually, the Spoken Dialog System (SDS) consists of four parts (in addition to the telephone interface):

1. the Continuous Speech Recognition (CSR) module,
2. the Natural Language Processing (NLP) module,
3. the Dialog Management (DM) module, and
4. the Text-To-Speech (TTS) module.

The system was connected to an ISDN line. Therefore, the input signals consist of 8 kHz 8 bit A-law coded samples. Feature extraction is done every 10 ms for frames with a width of 16 ms (Steinbiss et al., 1993, 1995). The first step in feature analysis is an FFT analysis to calculate the spectrum. Next, the energy in 14 mel-scaled filter bands between 350 and 3400 Hz is calculated. Apart from these 14 filterbank coefficients the 14 delta coefficients, log energy, and slope and curvature of the energy are also used. This makes a total of 31 feature coefficients.

The CSR uses acoustic models (HMMs), language models (unigram and bigram), and a lexicon. The lexicon contains orthographic and phonemic transcriptions of the words to be recognized. The continuous density HMMs consist of three segments of two identical states, one of which can be skipped. The output of the CSR module, and thus the input to the NLP module, is a word graph (Oerder & Ney, 1993; Ney & Aubert, 1994).

In the NLP module a stochastic attributed context-free grammar is used to parse this word graph. The main goal of the grammar is to find the information that is needed to perform the right query on the data base. Therefore, it is not necessary that all words are recognized and understood correctly. It is sufficient that important concepts (like e.g. origin, destination, and time of departure or arrival) are recognized correctly.

The DM module checks whether all information needed to perform a query on the data base is present (i.e. whether all slots are filled). If this is not the case, the system asks the caller explicitly for the missing information. When all slots are filled, the system accesses the data base.

The information found in the data base (and all other feedback mentioned above) is presented to the caller by means of speech synthesis. Language generation is limited to the concatenation of fixed phrases or by inserting the right words in open slots in carrier phrases. Speech synthesis is accomplished by concatenating pre-recorded phrases and words spoken by a female speaker.

3. BUILDING A DUTCH SDS

In order to build and train an SDS for a certain application, a considerable amount of data is needed. For collecting these data Wizard-of-Oz scenarios are often used. However, within the framework of the current project a different approach was chosen, which consists of the following five stages:

1. make a first version of the SDS with available data
2. ask a limited group of people to use this system and store the dialogs
3. use the recorded data to improve the SDS
4. gradually increase the data and the number of users
5. repeat steps 2, 3, and 4 until the system works satisfactorily.

3.1 The first version of the SDS

In Section 2 we provided a short description of the system developed by Philips Research Aachen. A first version of the SDS was obtained by localizing this German system for Dutch. How this was done is described in the present section.

CSR. The CSR component of the first version of the SDS was trained by using part of the Polyphone data base (Damhuis et al., 1994; den Os et al., 1995). This corpus is recorded over the telephone and consists of read and (semi-)spontaneous speech of 5000 subjects. For each speaker 50 items are available. Five of these 50 items are the so-called phonetically rich sentences, which contain all phonemes of Dutch at least once, while the more frequent phonemes occur more often. The five phonetically rich sentences of 500 subjects (i.e., 2500 utterances in total) were used to train the acoustic models of the first version of the CSR component. In the current Dutch version 38 monophones are used (see Table 1). Thirty-five of these models represent phonemes of Dutch, two represent allophones of /l/ and /r/, and one model is used for all non-speech sounds.

Table 1. The 38 monophones and some examples.

vowels				consonants					
i	l***ie***p	I	l***i***p	p	***p***ut	n	***n***at	s	***s***ap
e:	l***ee***g	E	l***e***g	b	***b***ad	l	***l***at	z	***z***at
a:	l***aa***t	A	l***a***t	t	***t***ak	r	***r***at	S	***sj***aal
o:	b***oo***m	O	b***o***m	d	***d***ak	L	ba***l***	j	***j***as
y	b***uu***t	Y	p***u***t	k	***k***at	R	ba***r***	x	li***ch***t
2:	d***eu***k	@	g***e***lijk	N	la***ng***	f	***f***iets	h	***h***ad
Ei	w***ij***s	u	b***oe***k	m	***m***at	v	***v***at	w	***w***at
9y	h***ui***s	Au	k***ou***d	n#	non-speech sounds				

Note that the data used to train the first version of the CSR are not application-specific. First of all, the data consist of read speech, and not spontaneous speech as in the intended application. Furthermore, the phonetically rich sentences of the Polyphone data base were selected from newspaper texts. Therefore, none of them are related to travels between two train stations.

The phonemic forms in the lexicon were taken from three different sources. Station names came from the ONOMASTICA data base (Konst and Boves, 1994), while the lemma forms of other words were taken from the CELEX data base (Baayen et al., 1993). The phonemic forms of words that were not found in these two data bases were generated by means of our grapheme-to-phone converter (Kerkhoff et al., 1984). Postprocessing was used to resolve systematic differences in the phonemic forms obtained from these three sources, i.e., to make the phonemic representations in the lexicon more homogeneous. For 42 words multiple pronunciations are included in the lexicon: 2 phonemic transcriptions for 40 words, and 3 phonemic transcriptions for 2 words. Of these 42 words 26 are station names, 10 are digits, and 6 are greetings. For all other words one phonemic transcription is present in the lexicon.

NLP. Since German and Dutch are rather similar from a syntactic point of view, for some parts of the NLP it was possible to make a direct translation from German to Dutch. However, in many other cases, such as time and date expressions, things appeared to be more complicated. First of all, each language has it own expressions for special days. For instance in Dutch we have “koninginnedag” (birthday of the queen, 30 April), which does not exist in German. Since it is common to say e.g. “de dag voor/na koninginnedag” (literally: the day before/after queen’s day), the system had to be taught to recognize expressions of this kind.

A second problem with the interpretation of time is related to the ambiguity of certain time expressions. The word “tomorrow” uttered in the small hours is a case in point. In order to interpret time expressions, the system uses the internal clock of the computer on which the NLP software runs. If the caller says “tomorrow”, the NLP adds 1 to the date of the internal clock. However, if the person calls at 00:10 and says “tomorrow”, (s)he probably means “today”, or, to be more precise, the current date of the internal clock. Therefore, this latter interpretation is adopted in the Dutch system.

Another date-related problem had to do with the fact that in the Dutch railway system there is a discrepancy between calendar days and ‘schedule’ days. A schedule day --which corresponds to the duration of the validity of tickets-- starts at 04:00 and continues until 03:59 of the next calendar day. The data base that provides the train timetable information in the Dutch dialog system adheres to this definition of schedule day. Therefore, all date-related expressions that are somehow connected to the time span between 00:00 and 04:00 had to be reconsidered. For instance, if a caller says “I want to travel today” at 02:00, the German system should interpret ‘today’ as the date of the internal clock, whereas the Dutch system should interpret ‘today’ as one day before the calendar date.

Furthermore, the interpretation of time intervals had to be adjusted too. The system allows queries for connections in a certain time interval. The start and end time of an interval must always be on the same ‘day’. If a ‘day’ covers the interval from 04:00 to 03:59, the conventional numeric order is not preserved. When a caller says “I want to arrive between six and three”, the present implementation allows for three interpretations, viz. 06:00 - 15:00, 06:00 - 03:00 and 18:00 - 03:00. In the original implementation only 06:00 - 15:00 would have been valid. Which of the three above-mentioned interpretations is chosen depends on the time of day when the query is received.

We were convinced that we could never figure out all the expressions Dutch people could use in order to get information about public transport just by introspection. At the same time, we did not have a large data base available that could be used to look for all possible expressions. Therefore, an alternative approach was adopted. A preliminary version of the grammar was made by translating and, where necessary, adapting the German grammar. This part of the SDS was then tested independently of the speech interfaces by using a keyboard version of the dialog system. A limited group of motivated people (about twenty) were asked to log in on the system and type their questions on a keyboard. The exact number of subjects who did participate is not known because the test was anonymous. The replies from the system appeared on the screen. Because people are likely to formulate their questions differently when they speak or type, the users were instructed to try to express themselves as they would do if they were speaking.

In this way we were able to test the grammar and to gather some text material that could be used to initialize the language model. In total 243 dialogs were obtained, consisting of 1 up to 6 queries. It turned out that the sessions of the users with this version of the NLP were extremely useful. On the basis of the log-files, many adjustments were made to the system. A nice example is that in the original German grammar there are 18 ways to give an affirmative answer and 7 ways to give a negative answer. On the basis of the log-files 34 affirmative answers and 18 negative answers were defined for Dutch.

DM. For the bootstrap version of the system the German DM was translated literally into Dutch. Some adaptations appeared to be necessary, though. For instance, the interface to the public transport data base had to be modified. Furthermore, some changes were required in the feedback to the caller. By way of illustration, in the German system train numbers are mentioned because these appear to be important for the caller. However, this piece of information is irrelevant in the Netherlands (people never refer to the train number) and was therefore excluded from the feedback in the Dutch system.

As mentioned above, a data base query is initiated only after all necessary information is available. Before an information item is considered as known and frozen, the caller is given explicit or implicit feedback about what the system thinks it has recognized. He can then disconfirm erroneous items and replace them with correct information.

TTS. Many adaptations had to be made to the speech output module of the system, because only the general approach from the German prototype could be copied. An inventory was made of the phrases that together form the questions and replies the system should be able to produce. Recordings were made of these utterances spoken by a female speaker. In the SDS these recorded utterances are concatenated to generate the speech output of the system.

3.2 Improving the SDS

The first version of the SDS was put in the PSTN in December 1995. The CSR module in this version was trained with DB0, i.e., the 2500 Polyphone utterances. Eighty employees of KPN (Royal Dutch PTT) received the telephone number of this system. The main criterion used to select them was that they had some experience with spoken dialog systems. However, they were not familiar with this specific SDS. They were requested to call it regularly, and their dialogs were recorded. In this way the data bases DB1 to DB3 in Table 2 were collected. Next, this telephone number was made known to 1200 other employees of KPN, and the data bases DB4 to DB7 were recorded. The data bases in Table 2 are built up incrementally, which means that DB2 is a superset of DB1, DB3 of DB2, etc.

Each utterance was orthographically transcribed. Out-of-vocabulary (OOV) words were detected automatically from the transcriptions. In this way words containing typing errors were also found and manually corrected. The OOV words were phonematized by hand and added to the training lexicon, so as to use all the collected data for training the system. However, not all new words were added to the recognition lexicon. Only those words considered to be relevant to the application were included in the recognition lexicon. Consequently, the size of the recognition lexicon gradually increased. Currently the recognition lexicon contains 1115 words.

Since words not present in the recognition lexicon can never be recognized correctly, it is important that the number of OOV words is not too large. As a measure we use the relative number of OOV words, which is calculated by dividing the number of OOV words for a data base by the total number of words in that data base. The relative number of OOV words as a function of the number of words in the data bases is shown in Figure 1. It can be observed that the relative number of OOV words is small. Apparently, we succeeded in making a bootstrap lexicon that contains most of the words used.

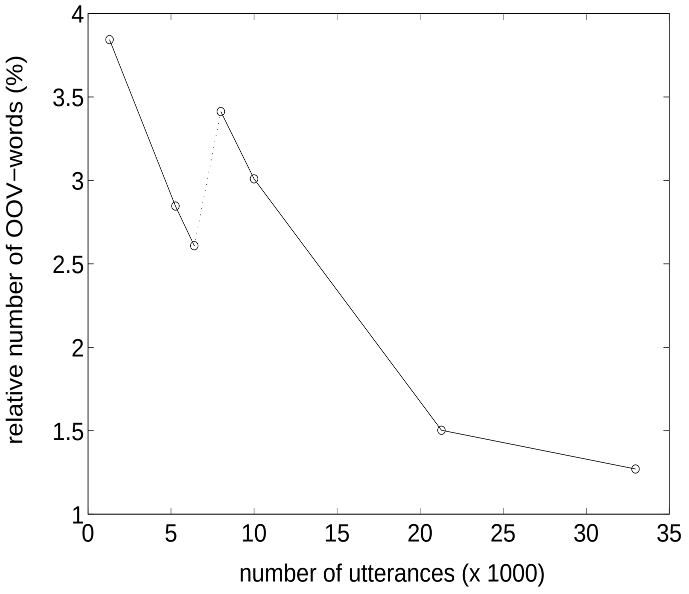

Figure 1. The relative number of OOV-words as a function of the number of utterances in the data base.

In Figure 1 one can also see that the relative number of OOV words decreases as the number of utterances increases from 1301 to 6401. In the beginning a fair number of OOV words are found. However, as the same group of people is likely to use more or less the same words to ask for information, the number of unknown words decreases gradually. After DB3 (6401 utterances) had been recorded, the telephone number of the system was made available to a larger group of people. It is conceivable that new people will use new words. As a matter of fact, the relative number of OOV words turns out to increase first and to decrease again later on (see Figure 1).

Table 2. Data bases used during the development of the SDS. Presented in the columns are the name of the data base, the number of utterances, the total duration of all speech signals (in min.), the total number of words, and the total number of distinct words.

Data bases	**# Utt.**	**Dur. (min)**	**# words**	**# distinct**
DB0	2500	282	28516	5951
DB1	1301	41	4575	370
DB2	5496	227	22167	678
DB3	6401	275	23279	683
DB4	8000	355	28197	785
DB5	10003	440	34587	845
DB6	21288	921	70773	1069
DB7	32971	1406	106880	1226

Whenever a sufficient amount of new data was collected, the lexicon was updated and language and phone models were trained again. The new models were compared to the old models (as will be described below), and those which performed best were chosen. In the on-line system the old models and the lexicons were replaced by the better ones.

In the early versions of the system we detected some syntactic constructions that were sometimes used by the callers, but were not handled correctly by the NLP. To improve the NLP, these syntactic constructions were added to the NLP's context free grammar. Furthermore, the NLP was trained with the same data used to train the language model (the bigram). During the training of the NLP the concept bigram model is constructed and the number of occurrences of syntactic units in the context free grammar is counted and stored in the NLP. As described above (see section 2), the concept bigram and the syntactic unit counts are used in deciding which parse of the word graph is chosen.

Although the first bootstrap version of the system was quite useful as a tool for data acquisition, tests performed recently show that some changes at the ergonomic level are required. For instance, the concatenation synthesis should be improved, information about complex journeys should be split into smaller chunks, and the caller should be able to interrupt the machine (barge-in capability). Some of these improvements in the DM module will be addressed in the near future.

3.3 The performance of the CSR module

Part of the data collected with the on-line SDS was kept apart as a test data base. Since we did not have much data in the beginning, the test data base had to be small: 500 utterances were randomly selected from the recorded dialogs. The consequence is that some of the utterances of a call are in the test data base, while the remaining part of the same call (and thus the same speaker) is in the training set. In total the test data base contains utterances from 465 different dialogs. The total number of words and characters in this test data base is 1674 and 9696, respectively. The total number of characters (or graphemes) can be used as a rough estimate of the total number of phones in the test data base.

The performance of the CSR module was evaluated for the whole word graph (WG) and for the best sentence (BS) obtained from this word graph. Both for the word graph and for the best sentence, word-error rate (WER) and sentence-error rate (SER) were calculated. In total this yields four measures that can be used for evaluation: WG-WER, WG-SER, BS-WER, and BS-SER. Because not everyone uses the same definition of WER it should be noted here that for calculation of the WER insertions are also counted as errors. Error rates for the word graph are obtained by searching the optimal path in the wordgraph, i.e., the path that matches the spoken utterance best.

In section 2 it was already explained that the NLP looks for specific concepts in the whole word graph, such as departure station, arrival station etc. Since these concepts are words, WG-WER would seem to be the most relevant evaluation measure. However, it is not necessary that the NLP recognizes every single word. Recognition of the above-mentioned crucial concepts will suffice. Although WG-WER is probably a better measure of the CSR performance than the other three indices mentioned previously, it is obvious that it is not an optimal measure. Indeed, the optimal measure would be a concept error rate for the word graph. In order to provide complete information about the performance of the CSR, the remaining three measures are also presented. The BS error rates give an idea of the quality of the phone models and bigrams, because the probabilities of the phones and bigrams are used to determine the BS from the WG. The SERs show how often the complete sentence is recognized correctly.

Table 3. Test-set perplexities and performance levels for the 4 different combinations of 2 phone models (P0 and P03) and 2 language models (L0 and L3).

System	**P0+L0**	**P03+L0**	**P0+L3**	**P03+L3**
perplexity	317.90	317.90	65.84	65.84
WG-WER	26.16	23.78	13.56	13.32
WG-SER	42.20	37.80	25.00	24.60
BS-WER	49.04	41.28	27.18	23.66
BS-SER	66.00	55.60	41.80	37.00

The different data bases were used to train language models (unigrams and bigrams) and the 38 acoustic models. Language models trained on data bases DBj will be called Lj. Phone models trained on data base DBn will be called Pn. In addition, phone models were trained on DB0 in combination with an application-specific data base DBm. These phone models will be called P0m.

The test data base was used to calculate error rates for various versions of the system (see Tables 3 and 4). First, phone models (PMs) and language models (LMs) were trained on the Polyphone material (DB0) alone. The resulting error rates are given in column ‘P0+L0'. DB1 was not used to train PMs and LMs because the number of utterances in this data base is too small. Since the error rates for DB2 are very similar to those for DB3, they are not presented here. Although various tests were performed, here we will present only the results of the tests in which the test lexicon consisted of the words contained in the test data base. The reason for this is that these results best illustrate the gradual changes in the performance of the CSR.

We first wanted to test how important application-specific data are for training PMs and LMs. In order to do so we calculated the error rates for all four combinations of two sets of PMs (P0 and P03) and two sets of LMs (L0 and L3). The results are given in Table 3. DB0 was not used in training the second set of LMs (i.e., L3), because this data base contains no utterances that are relevant to the present application. In this case using DB0 in addition to DBn would only worsen the LMs. For PMs, on the other hand, it appeared that PMs trained on DBn and DB0 are better than those trained on DBn alone.

The test-set perplexities in Table 3 show that the LMs trained on application-specific data (i.e., L3) are better than those trained on DB0 (i.e., L0), as expected. Using DB3 in addition to DB0 to train the PMs improves the performance (compare ‘P0+L0' with ‘P03+L0'). However, a much larger improvement is achieved if application-specific data are used to train the LMs (compare ‘P0+L0' with ‘P0+L3'). An additional, albeit relatively small, gain in performance can be obtained when both the PMs and the LMs are trained on application-specific data (compare ‘P0+L3' with ‘P03+L3'). On the basis of these results we may conclude that using application-specific data is more important for training the LMs than for training the PMs.

Above we stated that PMs trained on DBn and DB0 are usually better than those trained on DBn alone. If we compare ‘P03+L3' of Table 3 with ‘P3+L3' of Table 4 we notice that this is indeed the case for DB3. However, as the size of the application-specific data base DBn increases, it becomes less important to use DB0 in addition to DBn to train the PMs. For this reason, only tests in which application-specific data were used are presented in Table 4. In Table 4 it can be observed that the test-set perplexity and the error rates gradually diminish as the size of DBn increases. However, for DB7 there is hardly any further improvement.

Table 4. Test-set perplexities and performance levels for different phone models (Pi) and language models (Lj).

System	P3+L3	P4+L4	P5+L5	P6+L6	P7+L7
perplexity	65.84	50.30	48.20	35.24	35.22
WG-WER	14.81	11.89	11.35	8.48	8.42
WG-SER	25.80	22.80	20.40	16.60	16.80
BS-WER	26.11	21.45	20.61	16.79	16.37
BS-SER	37.40	32.20	30.20	25.80	25.60

3.4. The performance of the whole system

In the previous section we have only dealt with the performance of a single component of the total system, i.e., the CSR module. In order to obtain a complete picture of the performance of the whole SDS different types of measures must be collected. For three measures it was specified a priori what the criteria were that the system had to meet.

First of all, it was specified that the system should not need a cold start (reboot) more often than once per two months, a criterion that is easy to measure by keeping a system log. Our system needed only two reboots in six months time. Second, the response time of the schedule data base had to be less than 3 seconds. A powerful workstation was needed to meet this requirement.

Third, the success rate of the system had to be larger than 80%. Success rate is the percentage of inquiries in which the callers get the information they asked for. In order to measure the success rate of the system a formal test was carried out with 500 subjects, all of whom were employees of KPN. They were selected so as to cover all regional language varieties. Subjects did not receive specific tasks to carry out. Instead they were simply requested to make up one or more realistic questions, and then to try and get the answer to these questions from the SDS. In total 647 calls were made during the test, in which 890 inquiries were performed. All calls were orthographically transcribed and evaluated.

The median duration of an inquiry was 140 seconds. This is slightly longer than the median duration of the calls in the present operator service. Unfortunately, we have no data about the relative time spent in collecting the input information and in providing schedule information. This lack of detail applies to both the operator and the automatic version of the service.

Of the 890 recorded inquiries 153 had to be left out of the evaluation because they were not considered to be serious attempts to obtain information (e.g. people playing with the system, inquiries in which multiple speakers talked simultaneously, and inquiries in which non-existing station names were used). For the remaining 737 inquiries 94.7% proved to be successful. This is far beyond the goal of 80% successful queries that we had set ourselves.

4. DISCUSSION AND CONCLUSIONS

In this paper we have described the development of an automatic system for providing information about public transport in the Netherlands. Important characteristics of this system are that it was derived from a prototype that had originally been developed for German and that an alternative approach for collecting application-specific material was adopted, instead of the usual Wizard-of-Oz scenario.

This alternative method appears to have considerable advantages: First of all, no time is spent on the WOZ simulation. Instead, the real system is immediately realized. The whole application with all the modules is used from the beginning, and not just one component. In this way many practical problems pop up at an early stage, and can be solved before the final implementation takes place. Furthermore, the system used to collect the data is the real system and not a simple imitation. Finally, it is possible to collect speech material and to test, debug and evaluate the system at the same time.

However, one important disadvantage of this approach is that it requires that the first version of the system should work well enough to be used for data collection. We succeeded in making a suitable bootstrap for the following reasons. Firstly, because we could use the German prototype as a starting point. Secondly, because we had knowledge about German, Dutch, and this specific application. Thirdly, because German and Dutch are very similar. Fourthly, because speech data bases, albeit not application-specific, were available. Finally, because we used the data collected with the keyboard version. It is possible that under less advantageous circumstances, this approach would be less successful than it turned out to be in our case.

ACKNOWLEDGEMENTS

Part of the research was carried out in the framework of two projects: the MLAP project MAIS and the Dutch Priority programme ‘Language and Speech Technology’ which is funded by the Netherlands Organization for Scientific Research (NWO).

5. REFERENCES

Aust, H., Oerder, M., Seide, F. and Steinbiss, V. (1994). Experience with the Philips Automatic Train Timetable Information System. In: Proceedings IVTTA’94 2nd IEEE workshop on interactive voice technology for telecommunications applications (pp. 67-72). Kyoto, Japan.

Aust, H., Oerder, M., Seide, F. and Steinbiss ,V. (1995). A spoken language inquiry system for automatic train timetable information. Philips Journal of Research, 49 (4), pp. 399-418.

Baayen, R.H., Piepenbrock, R. and van Rijn, H. (1993). The CELEX lexical data base (on CD-ROM). Philadelphia, PA: Linguistic Data Consortium, University of Pennsylvania, USA.

Damhuis, M., Boogaart, T., in ‘t Veld, C., Versteijlen, M., Schelvis, W., Bos, L. and Boves, L. (1994). Creation and analysis of the Dutch Polyphone corpus. In: Proceedings International Conference on Spoken Language Processing (ICSLP) ‘94 (pp. 1803-1806). Yokohama, Japan.

Kerkhoff, J., Wester, J. and Boves, L. (1984). A compiler for implementing the linguistic phase of a text-to-speech conversion system. In: Bennis H. and W.U.S. van Lessen Kloeke (eds.), Linguistics in the Netherlands, pp. 111-117.

Konst, E.M. and Boves, L. (1994). Automatic grapheme-to-phoneme conversion of Dutch names. In: Proceedings International Conference on Spoken Language Processing (ICSLP) ‘94 (pp. 735-738). Yokohama, Japan.

Ney, H. and Aubert, X. (1994). A word graph algorithm for large vocabulary, continuous speech recognition. In: Proceedings International Conference on Spoken Language Processing (ICSLP) ‘94 (pp. 1355-1358). Yokohama, Japan.

Oerder, M. and Aust, H. (1994). A Real-time Prototype of an Automatic Inquiry System. In: Proceedings International Conference on Spoken Language Processing (ICSLP) ‘94 (pp. 703-706). Yokohama, Japan.

Oerder, M. and Ney, H. (1993). Word graphs: an efficient interface between continuous-speech recognition and language understanding. In: Proceedings ICASSP’93 (pp. 119-122). Minneapolis, USA.

den Os, E.A., Boogaart, T.I., Boves, L. and Klabbers, E. (1995). The Dutch Polyphone corpus. In: ESCA 4th European Conference on Speech Communication and Technology: EUROSPEECH 95 (pp. 825-828). Madrid, Spain.

Steinbiss, V., Ney, H., Haeb-Umbach, R., Tran, B., Essen, U., Kneser, R., Oerder, M., Meier, H., Aubert, X., Dugast, C. and Geller, D. (1993). The Philips research system for large-vocabulary continuous-speech recognition. In: ESCA 3rd European Conference on Speech Communication and Technology: EUROSPEECH ‘93 (pp. 2125-2128). Berlin, Germany.

Steinbiss, V., Ney, H., Aubert, X., Besling, S., Dugast, C., Essen, U., Geller, D., Haeb-Umbach, R., Kneser, R., Meier, H.-G., Oerder, M. and Tran, B.-H. (1995). The Philips research system for continuous-speech recognition. Philips Journal of Research, Vol. 49, No. 4, pp. 317-352.

Steinbiss, V., Tran, B. and Ney, H. (1994). Improvements in beam search. In: Proceedings International Conference on Spoken Language Processing (ICSLP) ‘94 (pp. 2143-2146). Yokohama, Japan.

Strik, H., Russel, A., van den Heuvel, H., Cucchiarini, C. and Boves, L. (1996). A spoken dialog system for public transport information. In: Proc. of the Dept. of Language and Speech, Nijmegen, Vol. 19, pp. 129-142.