An Automated Speech Processing System for Public Transport Information Services

R.J. van Vark[1,2] J.P.M. de Vreught[1,2] L.J.M. Rothkrantz[1,2]

Abstract.

The goal of a research project of OVR is to develop a system to automate part of its dialogues now held by human operators at its call centres. Operators at these call centres present information concerning Dutch public transport systems, ranging from local bus services to long distance trains. This paper describes the current development of such an ASP system and focus will be on the dialogue management part of that system.

1 Introduction

Last year OVR, a Dutch provider of information concerning national public transport, received over 10 million phone calls at its call centers and this number is rising rapidly year after year. Although the system operates only during day time, clients expressed a desire for this information during night time as well. To handle the total increase in demand OVR has started a research project to investigate the usability of Automated Speech Processing (ASP) in their company. OVR is involved in European projects on the retrieval of information concerning public transport like Mais, Railtel, and Arise.

Several universities and large companies in the Netherlands are currently working on building an ASP system fo OVR. The major problems with current systems are that speech recognition is still in its infancy and the dialogues are missing mixed initiative, i.e. the system asks all questions. OVR has chosen to take an evolutionary approach for creating their ASP system. The current evolution, VIOS2, is capable of handling free speech (using a limited vocabulary), but the dialogue structure is still far from optimal.

Our Alparon project aims to create a Dialogue Manager (DM), based on a corpus covering 115 hours of speech [7], consisting of over 5000 dialogues made at the call centers of OVR. After transliteration we decided to code about 500 of them for further analysis. Based on our coding scheme [9] we were able to make a classification of the dialogue complexity with respect to the type of ASP system needed for such dialogues.

At a side track we have investigated the human-human versus human-computer interface in an experiment where the machine is secretly simulated by another human. During part of this Wizard of Oz experiment 400 people were asked to use the simulated system and the human operators for retrieving information on the same queries.

The emphasis of the paper will be on the corpus based DM we are currently working on. The frontend consists of a speech recogniser (not implemented by us), a parser, and a coder. The coder transforms a parse graph into a set of recognised dialogue acts (rda's). Two of the responsibilities of a DM are to

[1]This research was funded by Openbaar Vervoer Reisinformatie (OVR) and Senter.

[2]Delft University of Technology, Faculty of Technical Mathematics and Informatics, Julianalaan 132, 2628 BL Delft, The Netherlands, alparon@kgs.twi.tudelft.nl

determine which dialogue act is most probable and what is the best response to this dialogue act. The backend will transform this in a textual representation and finally a speech synthesizer (not implemented by us) will answer the client with a spoken reply.

The DM has to fulfill several tasks. The reasoning and deduction system is the brain of the DM and is an expert system that drives the dialogue by performing subtasks as dialogue selection, updating of information system-wide, response generation, dialogue act prediction, and phrasing database queries. The black board manager manages, remembers, and updates all relevant information for the DM. There are several black boards: an information board, a control board, a history board, a state board, and a travel advice board. The final task in the DM is to interact with the large OVR database.

Following the introduction, related work is discussed first, followed by the analysis of our corpora. Next an overview of the ASP system is described. The fifth section covers the dialogue manager in more detail as dialogue management is thought to be essential in the OVR environment and in ASP in general. Finally, future research and developments are presented.

2 Related Work

The first stage of our project was directed to the analysis of our corpus. Carletta et al. [3] developed a coding scheme for task oriented dialogues existing of three levels. At the highest level dialogues are divided in transactions or subdialogues corresponding to major steps in the participants' plan for completing the task. At the intermediate level conversational games consist of initiations and responses. A conversational game is in fact a sequence of utterances, starting with an initiation and ending with the requested information being transferred. At the lowest level conversational moves are utterances which are initiations and responses named according to their purpose. Our coding scheme has been strongly influenced by their scheme, but it is, however, more hierarchically oriented. This results in a more general coding at the top level, while more detail can be obtained at deeper levels [9].

Veldhuijzen van Zanten [10] describes the architecture of OVIS2, a demonstrator system that handles spoken language dialogue concerning time-table inquiries of the Dutch Railways and it is incorporated in the VIOS system. OVIS2 is an improved prototype based on OVIS1 which was a translation of a similar German prototype [2, 5]. In OVIS2 a central discourse representation is maintained during the dialogue. This datastructure respresents everything said so far. All utterances of both the system and client are represented on several abstraction levels: a syntactic, semantic, pragmatic, and information level. We intend to move large parts of such a discourse representation into our dialogue manager as all information is coded in our moves and their informational subcodes. Almost all information is only relevant to the dialogue manager. Information relevant to other components of our system will be stored in a central 'linguistic/atmosphere' database (see section 4).

In the VERBMOBIL project [1] researchers developed a taxonomy of dialogue acts. To model the task oriented dialogues in a large corpus they assumed that such dialogues can be modelled by means of a limited but open set of dialogue acts. They defined 17 dialogue acts and semi-formal rules to assign them to utterances. These assignment rules served as a starting point for the automatic determination of dialogue acts within the semantic evaluation system. From the analysis of an annotated corpus of 200 dialogues they derived a standard model of admissible dialogue act sequences. They use four subsystems to detect dialogue acts automatically: speech recognition, syntactic analysis, semantic analysis/evaluation and dialogue processing. We use similar subsystems to detect valid dialogue acts and dialogue act sequences.

3 Analysis of the Corpora

The first steps toward an automated system were to analyse the existing system and to determine what kind of dialogue form is appreciated by most users. For the first task we had a corpus of 5205 dialogues covering 115 hours of speech at our disposal [7]. The dialogues were recorded calls made by users to the call centers of OVR. The large size of the corpus covers the OVR domain very well.

Recording of these dialogues took place from May 1995 to July 1995 at 9 different call centers. These recordings were made by the Dutch PTT using incoming calls at the call centers of OVR. For privacy reasons the recordings were deprived of all personal information like names and addresses. Also the period spent on hold was deleted from the recordings.

In table 1 the frequency table is given for all conversations. Actually, the dialogues were grouped in four categories: train information (40% of all dialogues), bus/tram information (32%), train & bus/tram information (12%), and miscellaneous (16%). The median of the dialogues containing train information only is approximately $\frac{3}{4}$ minute while from all other categories the median lies around $1\frac{1}{4}$ minute.

As a second step we analysed a sample of 483 dialogues out of this corpus. In this sample utterances with the same intensions were coded similarly [9]. The coded dialogues enabled us to take a closer look at the overall structure of those dialogues without being hindered by all sorts of minor variations

Table 1: Dialogue duration

duration (sec.)	amount #	%	duration (sec.)	amount #	%
0- 10	1	0.0	105-120	283	5.4
10- 20	122	2.3	120-150	294	5.6
20- 30	590	11.3	150-180	185	3.6
30- 40	740	14.2	180-210	118	2.3
40- 50	677	13.0	210-240	69	1.3
50- 60	504	9.7	240-270	55	1.1
60- 75	670	12.9	270-300	29	0.6
75- 90	479	9.2	300-360	30	0.6
90-105	326	6.3	360- ∞	33	0.6

('good morning', 'good day', and 'hi' are all greetings: at this stage there is no real need to distinguish between them). The coded sample provides a good insight into the dialogue structures actually occurring in the human-human interface [7, 9].

Based on our coding scheme we were able to make a classification of the dialogue complexity with respect to the type of ASP system needed for such dialogues. In 28% of the dialogues the user was able to present all necessary information without any interference of the operator. In 60% of the cases the operator had to solicit further information of the clients, while in 4% the operator actually helped the client by introducing information needed to formulate his query. In only 1% of the dialogues the client's query could not be answered.

At a side track a Wizard of Oz experiment was conducted to investigate human-human versus human-computer interfaces [7]. During a Wizard of Oz experiment the computer is secretly simulated by a human. In our experiment 400 persons were asked to call both the simulated system and the human operator to retrieve information about the same queries. Using such a scheme human-human and human-computer dialogues can be compared objectively. Additionally the 400 persons were asked to perform different tasks with the simulated system as well.

During the experiment the following six variables were manipulated:

- full verification of departure/arrival time/place versus no verification,
- presentation of the travel advice as a whole or in little chunks,
- a three second delay after each utterance of the system versus no delay,
- the system makes mistakes in the recognition of names versus a near perfect system,
- a female versus a male voice,
- a non-directive versus a directive dialogue style.

Although the analysis of this experiment is still ongoing, some results are already known. Most important was that the making of mistakes by the system was very much disliked. The possibility of a non-directive dialogue structure was highly appreciated as was the presentation of the travel advice in little chunks. The other variables were of less influence on the appreciation of the ASP system.

4 The ASP system

In this section the overal structure of our system will be described. It consists of several parts (see figure 1), which each serve a specific goal. The different parts will be described below.

4.1 User Interface by Telephone

The interface between user and automated speech processing system takes place by telephone. Data resulting from this interface is continuous speech coded using a-law, the European coding standard for telephone communications. Data will be presented to this interface by applying the same coding standard.

4.2 Speech Recogniser

The speech recogniser takes care of transcribing a continuous speech signal into text and is divided into several subsystems. First, speech has to be detected, i.e. the speech recogniser has to deduct when the client is speaking and when the client is silent. Secondly, speech fragments have to be segmented to apply speech recognition. We intend to investigate the use of methods presented by Seligman et al. [8]. Finally, speech recognition itself has to be applied to speech fragments. Output of the speech recogniser takes the form of n-best sentences: the n sentences with the highest probability of having been uttered by the client.

4.3 Parser

The parser takes n-best sentences as input and checks these sentences for grammatical correctness. Based on this grammatical correctness of the input sentences a parse graph is created. This parse graph contains all possible combinations of uttered parses as derived from the n-best sentences. Information feedback provided by the dialogue manager can be used to select valid sentences to create the correct parse graph. Due to the current level of technology in speech recognition and the nature of spoken language, robust parsing will have to be applied [4]. Using robust parsing failures during the speech recognition process, like misinterpreted or unrecognised words, can be corrected.

Besides creating parse graphs from input sentences, the parser stores characteristics

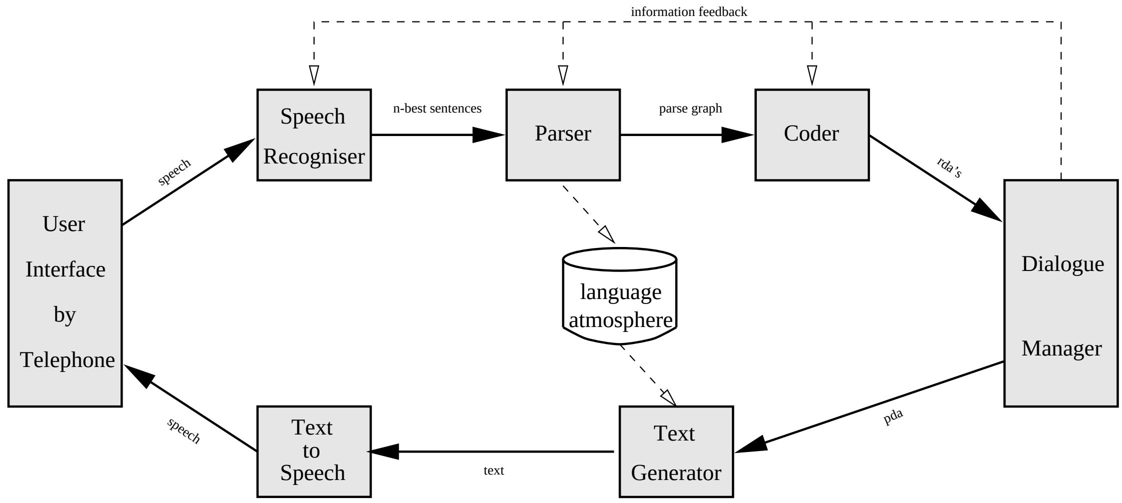

Figure 1: Overall structure

of the language the client uses in the linguistic/atmosphere database. This database can later be used by the text generator to make minor adaptations to the produced text in order to make the client feel more comfortable with the system. Possible characteristics are the use of certain words used when being polite and the atmosphere in which the conversation proceeds.

4.4 Coder

This is the part which differs from most other models for automated language processing, i.e. the location of the coder in the structure. Most systems use annotated sentences, typically produced by a parser, as input for their dialogue manager. Sometimes a somewhat higher coding of small groups of words is used. The dialogue manager then derives which dialogue act was meant by the client. Using information based on recent dialogue acts, we think we can recognise nearly all dialogue acts occurring in natural dialogues. Output of the coder is a set of recognised dialogue acts (rda's) with a probability above a certain threshold. The dialogue manager will eventually decide which dialogue acts is the most probable.

4.5 Dialogue Manager

The dialogue manager takes care of controlling the dialogue. The dialogue manager consists of several subsystems, like dialogue act prediction and querying the database. The dialogue manager is the core of the system and is described in more detail in the next section. Output of the dialogue manager is a proposed dialogue act, which is relayed to the text generator.

4.6 Text Generator

The text generator transforms proposed dialogue acts into grammatical correct sentences. Besides background knowledge of domain and language, the generator uses information stored in the linguistic/atmosphere database to adapt text to the client's way of speaking. Hopefully, this will increase the acceptation of the system. However, adaptation of text towards the user is an issue which we will implement until later in the development stage.

4.7 Text to Speech

The final part of the system transforms text into spoken words. Output of this part is a sound signal which is presented to the user interface, i.e. a telephone interface using the a-law standard coding scheme.

5 Dialogue Manager

The dialogue manager is the core of the automated speech processing system. This subsystem takes care of managing human-computer dialogues and is therefore rather complex. On the highest level, the dialogue manager consists of 3 different subsystems as can be seen in figure 2. Each of the subsystems black board manager and reasoning and deduction subsystem can be further subdivided.

5.1 Reasoning and Deduction Subsystem

This subsystem takes care of the largest part of intelligent processing in the dialogue manager. Many different tasks can be distinguished in the dialogue manager. Many of these tasks of which some are described below, can be executed simultaneously.

Dialogue Act Selection The first task of the dialogue manager is to derive the most probable dialogue act from the set of dialogue acts generated by the coder subsystem. This selection of the most probable dialogue act is done by the dialogue manager and not by the coder, because the dialogue manager can use information concerning earlier dialogue acts to help selecting the best dialogue act. In many cases, the list of recognised dialogue acts shall be very short, often only containing one element.

Updating of Information Once the correct or most probable dialogue act has been selected, information conveyed by this dialogue act has to be stored on all black boards. This is of course done by using the black board manager. Besides storing the information, also the set of recognised dialogue acts is stored on the history board in case repairs have to be initiated in future.

Response Generation Using information stored on the black boards and knowledge integrated in the reasoning algorithms, the best response on the user's dialogue act can be generated. This proposed dialogue act (pda) contains information codes when necessary and is sent to the next subsystem which will transform this dialogue act into text.

Dialogue Act Prediction One of the tasks which can be performed during 'idle time', is the predicition of future dialogue acts. This prediction can be used to aid speech recognising, parsing and coding of user utterances. When only a limited number of dialogue acts can be expected, these subsystems can be focused on keywords and specific word combinations occurring in these dialogue acts. Dialogue act prediction can be done using n-gram dialogue act probabilities [6], which method has been adapted from speech recognition.

Phrasing Database Queries Another important task of the reasoning subsystem is phrasing database queries as soon as enough information slots are defined. Phrasing this query and presenting it to the database manager can also be done during 'idle time', while the system checks with the user to see if slots are filled correctly. A special query language can be used invariant of the language used to query the OVR database, because the database manager will translate the queries between the two subsystems.

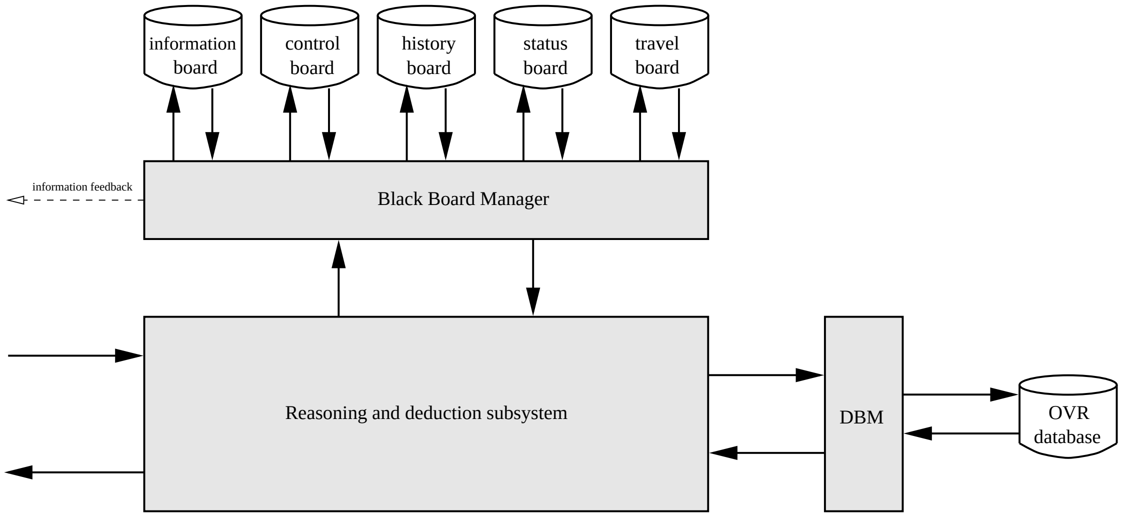

Figure 2: Overview of the Dialogue Manager

5.2 Black Board Manager

All information concerning the current dialogue are stored on black boards, i.e. the black boards make up the memory of the dialogue manager. The black board manager takes care of keeping the information on all black boards up to date. Changes to values on all boards are remembered as well as negative information, i.e. information which has been negated ('No, I do not want to go to Amsterdam'). Based on values and mutations on these values the board manager can produce the most probable value for all information slots found on the black boards when these are required by the reasoning and deduction subsystem.

Information board This board contains values stored in information slots, like departure place, arrival time, and transport modality (train, bus, etc.). These slots are necessary to retrieve relevant information from the OVR database.

Values for these information slots have been supplied by the client during the conversation and have been accepted by the dialogue manager. Old values which have been changed by the client and the way these changes have taken place or have been interpreted are also stored on the information board.

Algorithms are necessary to check values stored in different slots for consistency. These checks can be used to detect misinterpretations of client utterances. For instance, in case the client has one hour to travel from Amsterdam to Groningen, a repair can be initiated because the journey from Amsterdam to Groningen takes at least two hours. These algorithms are also necessary to present valid information to the reasoning subsystem.

Control board The control board is used to make sure the dialogue has a more or less standardised structure. Goals are stored on this board ordered by importance. A goal can be deleted from the control board when it has been fulfilled. When a goal is next to be handled, it can be replaced by several subgoals necessary to fulfill the entire goal. By changing the expert systems used to manage the control board and the goals put on the control board, the structure of the resulting dialogues can be altered. Experiments will have to be undertaken how the control board can best be implemented to match human-human dialogues in the most acceptable fashion.

Examples of possible goals are introducing the system to the client, getting information from the client to retrieve information, and presenting the retrieved information to the client. Getting information from the client can be split in several subgoals, for instance getting the departure place and transport modality.

History board The history board contains the history of the dialogue. The history consists of both dialogue acts uttered by the client as dialogue acts produced by the dialogue manager. The history of the conversation can be used to predict future user dialogue acts. When the client tells the system of a misunderstanding, the history can be used to detect where this misunderstanding originated and derive plausible repair actions. Also, labels can be attached to misunderstood dialogue acts, in order to prevent future mistakes originating from the same misinterpreted dialogue act.

State board This board consists of slots describing the current state. The state consists of 2 parts:

- state of the dialogue, for instance dialogue phase, state of the information slots (complete/incomplete), and possible deadlocks in the interaction between client and system,
- state of the ASP system, for instance a slot which states the OVR-database is off-line when no connection can be achieved to this database or a slot which indicates the number of clients in queue waiting to obtain travel information.

Travel board The travel board contains the complete travel advice once an advice has been retrieved from the OVR database. The entire schedule is stored first instead of presented directly to the user, because such a travel schedule can be rather complex and is usually broken down in small pieces. These small pieces can be easily handled by the client and can be individually acknowledged.

5.3 Database Manager

The database manager takes care of the interaction between ASP system and OVR database. Using an intermediate database manager results in a more flexible interface as retrieval commands from the system have to be translated in query commands of the database. When the interface to the database changes, only this intermediate interface has to be adapted. Also the dialogue manager does not suffer from a slow connection or high load on the database server. When a travel advice is retrieved from the database, it is stored on the travel board using the black board manager.

6 Future Research

The goal of the current research project is an ASP demonstrator for information retrieval dialogues. To reach this goal we have planned several subgoals along the way.

Based on a corpus of over 5000 human-human dialogues a coding scheme has been developed for analysing information retrieval dialogues. However, the coding scheme was developed for analysing dialogues and not for recognition. The coding scheme has to be adapted slightly for recognition, which will be our first goal as all dialogue management tasks depend on the coding scheme.

Analysis of this corpus and the Wizard of Oz corpus produces good insight into how an ASP system for public transportation information dialogues should interface with its users. Following adaptation of our coding scheme, we will implement several subsystems of the dialogue manager. Implementation of the black board manager and prediction of dialogue acts will be among the first systems to be implemented.

Simple prototypes of the parser and the coder have been developed. However, these prototypes were only pilot studies based on a small subset of the dialogues. Therefore, it will take some time until these subsystems will have matured into prototypes with an accuracy high enough to be used in the demonstrator.

In the final step we will try to fit all into a prototype that will show the strengths of the techniques used. If this prototype works satisfactory, it will be reimplemented more efficiently and in parallel by others such that it can be used by OVR. It is hoped that the ASP system will supplement the current operators such that their workload can be reduced.

References

[1] J. Alexandersson and N. Reithinger. Designing the dialogue component in a speech translation system, A corpus based approach. In *Corpus-based Approaches to Dialogue Modelling*, 9th Twente Workshop on Language Technology, pages 35–43. University of Twente, 1995.

[2] H. Aust, M. Oerder, F. Seide, and V. Steinbiss. Experience with the Philips automatic train timetable information system. In *Proceedings of the IVTTSA '94*, pages 67–72. Kyoto, Japan, 1994.

[3] J. Carletta, A. Isard, S. Isard, J. Kowtko, G. Doherty-Sneddon, and A. Anderson. The coding of dialogue structure in a corpus. In *Corpus-based Approaches to Dialogue Modelling*, 9th Twente Workshop on Language Technology, pages 25–34. University of Twente, 1995.

[4] R. Kiezebrink and J.P.M. de Vreught. An annotated bibliography on robust parsing and related topics. Technical Report CS 96-58, Delft University of Technology, 1996.

[5] M. Oerder and H. Aust. A real-time prototype of an automatic inquiry system. In *International Conference on Spoken Language Processing*, pages 703–706, 1994.

[6] N. Reithinger and E. Maier. Utilizing statistical speech act processing in VERBMOBIL. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL-95)*. Cambridge, 1995.

[7] L.J.M. Rothkrantz, R.J. van Vark, J.P.M. de Vreught, J.W.A. van Wees, and H. Koppelaar. Automatische spraakherkenning en OVR-dialogen. Technical report, Delft University of Technology, Knowledge Based Systems, 1996. in Dutch.

[8] M. Seligman, J. Hosaka, and H. Singer. "Pause Units" and analysis of spontaneous Japanese dialogues: Preliminary studies. In *Workshop Dialogue Processing in Spoken Language Systems*, European Conference on Artificial Intelligence, 1996.

[9] R.J. van Vark, J.P.M. de Vreught, and L.J.M. Rothkrantz. Classification of public transport information dialogues using an information based coding scheme. In *Workshop Dialogue Processing in Spoken Language Systems*, European Conference on Artificial Intelligence, 1996.

[10] G. Veldhuijzen van Zanten. Pragmatic interpretation and dialogue management in spoken-language systems. In *Dialogue Management in Natural Language Systems*, 11th Twente Workshop on Language Technology, pages 81–88. University of Twente, 1996.