Building Dialogue Systems that Sell

Stan P. van de Burgt+, Toine Andernach*,
Hans Kloosterman+, Rene Bos*, Anton Nijholt*

+KPN Research, PO Box 421, Leidschendam, the Netherlands.
*University Twente, Computer Science, PO Box 217, Enschede, the Netherlands.
Email: NLP@research.kpn.com

Abstract

This paper addresses the application of dialogue management techniques in a buyer-seller situation. In particular, we will describe the techniques used in the SCHISMA system, a system that provides the caller with information on theatre performances, and optionally "sells" the caller one or more tickets for a given performance.

1 Introduction

Interactive voice response (IVR) technology is applied successfully in many services, such as:

- information services
- call routing services
- confirmation services
- ordering services

When applied in the telephone network, these services are often build using the DTMF tones of the telephone. In recent years, spoken commands were added using speech recognition techniques for isolated (though speaker-independent) word recognition.

The current state of the art allows the recognition of words within a sentence, or even (again speaker-independent) recognition of the sentence as a whole.

As this technology advances commercial application of natural language processing (NLP) technology in IVR services becomes more viable.

Contrary to what many researchers argue, the major advance in NLP that will render commercial applications viable, is not more efficient parsing or thorough semantic analysis. The main factor will be our ability to build systems that can take part in a friendly, efficient and co-operative dialogue. Dialogue management techniques are thus a key factor in successful IVR systems.

This paper addresses the application of dialogue management techniques in a buyer-seller situation. In particular, we will describe the system SCHISMA (ter Doest 1995; Andernach 1995) that is being developed in a co-operative project of Twente University and KPN Research in the Netherlands. The system provides the caller with information on theatre performances, and optionally sells the caller one or more tickets for a given performance.

The SCHISMA system is an evolving "system-in-the-loop". It started as a pure, human driven, *Wizard of Oz* (WoZ) system to which several additions are made as time went by and more subsystems became available. This way, the system evolves to the final system we envision. Currently parts of the generator

and the dialogue management modules are incorporated, while pre-processing modules are being incorporated later this year.

2 Getting to know the user

Several observations in the literature confirm our belief that users have certain expectations of the behavior of the computer they communicate with. They dislike too elaborate and too polite utterances of a computer. Muller et al. (1977) for example, found out that the elliptical style of the system turned out to be an improvement over previous, simulated versions where more elaborate speech was found to be somewhat irritating. Van Katwijk (1979) found that friendly and polite utterances were not always appreciated. Some subjects remarked that their courtesy was wasted on a machine when they reacted with a "goodbye" to a "goodbye" utterance of the system.

Dahlback & Jonsson (1986) stress the importance of giving a correct model of the system's structure and capabilities to the user; the user must not be given the unrealistic impression of the 'intellectual' competence of the information system, caused to a sophisticated and varied linguistic behaviour of the system.

On the other hand, people tend to adapt their use of language to their dialogue partners too (Dahlback, 1995; Zoltan-Ford, 1991). This holds for human dialogue partners as well as for computers; as Morel (1989) pointed out, computer-like voice and behaviour do have influence on the linguistic behaviour and the prosody of the users of a dialogue system.

From the previous observations we could conclude that a machine's linguistic behaviour has a large influence on the user's linguistic behaviour and we cannot merely base the design of our dialogue system on the analysis of human-human dialogues.

We follow Dahlback & Jonsson (1986) in the assumption that the construction of a natural language dialogue interface will have succeeded if the user feels comfortable with it. If subjects use a restricted language and simple dialogue phenomena without getting uncomfortable this will be acceptable.

However, given the current state of the art of natural language man-machine communication, it is still unfeasible to investigate human behaviour in fully integrated natural language understanding systems in a real environment, as such systems are not available. Therefore, we should look for alternative techniques; one of the most common techniques adopted for the design of man-machine interfaces is the elicitation of man-machine dialogues where the role of the machine is simulated. In these so-called Wizard of Oz-experiments subjects interact with a machine without knowing that the turns of the machine are simulated by a so-called *wizard*. This kind of experiments can be of great value for the design of a dialogue system because of the insights we can gain in the (linguistic) behaviour of people while they are talking with machines.

We use the following procedure for developing a dialogue system, a combination of a procedure proposed by Fraser (1995) and a "system in the loop" approach:

1. Study the human-human dialogues in the domain for which the system is to be developed.
2. Define WoZ simulations.
3. Conduct WoZ experiments
4. Transcribe the resulting dialogues
5. Specify the dialogue system
6. Implement a version of the dialogue system

7. Test the system (developers)
8. Test the system (real users)
9. Use data from 7 and 8 to improve the system
10. If many modifications are needed, perform new WoZ experiments
11. Return to step 6

3 Dialogue management techniques

This section describes several techniques that are (being) implemented in the SCHISMA system.

3.1 Pre-processing of User Utterances.

As a first step, the utterances are analysed for cue words and domain-dependant constructs like:

- Prepositions
- Numbers
- Expressions of date and time
- Domain dependant names that are listed in the database
- Wh-words
- Yes/No expressions
- Surface structure

Depending on the input mode, speech or keyboard, additional processing is possible. With typed input, additional morphological pre-processing is performed. After this, we perform simple grammatical analysis in order to extract the basic noun phrases and the possible relations between them. The results of this pre-processing step are collected in a information frame (see section 3.3).

3.2 Statistical Techniques

The function of an utterance in a particular place in the dialogue, cannot trivially be concluded from the utterance itself. Combining certain features of the current and previous sentences, however, could hold the clue to this function.

We use statistical techniques to classify the utterances according to their function. One way of doing this is to annotate a corpus of dialogue utterances and use the result to train a classifier. This approach, however, relies heavily on people's subjective judgements which appears to be unreliable for the tasks of making a class hierarchy and finding class assignment rules.

In a our approach we try to prevent this as much as possible and aim at automating the classification process. In this approach, the Wizard of Oz corpus was our empirical basis, in (partly) being both training and test corpus. We assume that there is a strong relationship between form and function of utterances; form is crucial for determining function (see Hinkelman and Allen, 1989). Relevant formal aspects of utterances (also called cues) are utterance type (or "mood") and syntactic category. Other aspects are the grammatical subject of utterances, the type of verbs and the presence of question marks and wh-words.

Before the actual classification, every utterances in the Wizard of Oz corpus was represented by a vector of cues, a cue pattern. To automatically derive classes from this set of cue patterns, we applied an unsupervised classification algorithm. This algorithm allowed us to find the most probable set of classes based on statistical analyses of the cue patterns.

Unsupervised classification yielded a number of useful classifications. Results showed that some classes are stronger than other classes and that some cues are stronger than other cues. This information can be used for determining the set of classes in the eventual system and for testing with new sets, thereby grouping classes and omitting weaker cues.

Then, the output of the unsupervised classification process (cue patterns with their most probable class) was randomly split in a training set of 75% and a test set of 25%. The training set served as input for a supervised classification algorithm which resulted in a set of rules for describing the relation between the cue patterns and their classes.

Application of the supervised algorithm yielded a set of 44 rules for deriving classes for 206 different cue patterns. Below you find two examples of these rules:

```
IF SUBJECTTYPE = 2nd pers. Pron.

AND QUESTIONMARK = yes THEN CLASS = 2 [0 0 96 0 0 0 0]


IF UTTERANCETYPE = N/NP(s) THEN CLASS = 0 [109 0 0 0 0 0 0]
```

The first rule expresses that class 2 is assigned to utterances with a second person personal pronoun and a question mark. The second rule expresses that class 0 is assigned to utterances which consist of (a number of) Ns or NPs.

Accuracy of the rule set was tested on the test set. Accuracy of the rules for a certain class can be expressed by the ratio of the number of well-predicted cue patterns and the total number of cue patterns in that class. This way of testing yielded an accuracy of at least 90% for every class.

Note that finding and deriving classes of utterances with the method described is a multi-pass, iterative process; training and testing with different cues and class-sets is necessary to find optimal classifications. The classification results described are promising; classes yielded by the unsupervised classification algorithm are useful for our dialogue system and the information about the relative influence of cues and classes generated by this algorithm are useful for finding optimal sets of classes and cues. The supervised algorithm allowed us to derive a sufficiently general and accurate set of classes and cues.

3.3 Dialogue Rules and Procedures

The dialogue manager (DM) is responsible for maintaining the flow of conversation. After each utterance of the user (or after a time-out), the DM combines extracted features from the most recent utterance with the dialogue state, optionally consults the database, and decides on the topics and parameters of its next utterance. These steps are defined in dialogue rules, a dialogue state diagram, and dialogue procedures. Baekgaard (1995) uses FSAs as a basis for dialogue management. One of the more important drawbacks of a dialogue manager based on a FSA (FSA-DM), is its rigidity, a property also found by Alexandersson et al. (1995) and Aust and Oerder (1995). The FSA-DM is unable to cope with small and major deviations from the prescribed paths. Alexandersson et al. proposed a repair mechanism to overcome this unwanted property.

One other possible extensions one could propose is the addition of recursive operators which enable the

dialogue designer to mark specific points in the dialogue where a subdialogue, specified in another FSA, may be initiated. This way, the FSA is implicitly extended with a stack on which the state before entering a subdialogue is temporarily stored. The result is similar to a push down automaton (PDA) or a recursive transition network (RTN) which have the power to recognise context free languages.

This extended machinery is, as we and other scholars found, still too rigid to model dialogues that occur in real live or, as in our case, in the SCHISMA corpus of WoZ dialogues.

A major omission is the inability to branch to earlier, possibly "completed" dialogues. Another problem that is hard to solve elegantly with PDA based machinery, is the question of were to branch to, if at all, given the next utterance of the interlocutor.

Inspired by the stack of the PDA approach, and some of the dialogues found in the SCHISMA corpus, we came to use an *activation list*. An activation list is an ordered list of frames. Each frame reflects some earlier state in the current dialogue, just after the system computed its utterance.

The frame holds the domain information collected in the dialogue sofar (names of plays, actors, groups, dates, ...), as well as some information on the dialogue as well.

Initially, just after the first prompt in the dialogue is uttered, the activation list contains one empty frame. Each time an user utterance is received, the list is searched, top to bottom, to find a frame that is relevant to the current user utterance (see below). This frame is then updated with the user's utterance after which it is copied to the top of the list. Based on the updated frame, the system's utterance is computed, which is subsequently uttered. More schematically:

```
 eod = false;

 prompt(INIT);

 al = emptyframe();

 while (eod(al)) {

u = get_utterance();

if (consistent(u, db) {

nf = copyframe(findmatch(u, al));

update(nf, u);

if (consistent(nf, db) {

al = insert(nf, al);

sf = systemutterance(al, u);

al = insert(sf, al);

}

}

 }
```

How can a matching frame be identified? While searching the activation list top to bottom, each field in the frame is compared to the relevant part of the user's utterance. A frame matches if no part of the utterance conflicts with the relevant field value. This way the most recent frame is found that still is valid given the current utterance. Because the found frame is subsequently copied to the top of the activation list, it thus becomes the current focus. More details on this subject can be found in (Bos, 1996).

3.4 Generating System Utterances

After the dialogue manager established the topic(s) and parameters to be conveyed in the next turn, the systemÕs utterance is computed. For this purpose we are using a semantic grammar1 that is domain dependant. The actual realisation depends on the state of the dialogue, including the mutual knowledge. A simple example is shown below. Here a (flattened) rule of the semantic grammar lists three parameters, all shown in upper case.

The performance of PERFORMER [named "TITLE"] can be attended on DATE*

The first parameter, PERFORMER is mandatory, that is, if this parameter is not uniquely defined in the current dialogue state, this rule can not be used to compute an utterance. Next, the TITLE is, together with the prefix "named", listed between brackets. This indicates that this part is optional: If it is undefined in the dialogue state, or if it is "mutual knowledge", i.e. it is mentioned before, it will not be realised. The third and final parameter is DATE. This parameter is mandatory, but could be multi-valued, as indicated with the *symbol. The actual realisation of this parameter is specified in a more specific rule.

4 Conclusions

The SCHISMA system is an evolving "system-in-the-loop" on the domain of theatre performances. Besides providing information on the current performances, it enables the user to buy tickets for these performances. In this paper we showed the current status of this system, together with the latest developments that will be integrated as subsystems in the next months. Especially the results of the classification experiments look promising and will, together with the dialogue management procedures, bring the system closer to its final goal: a stand-alone dialogue system.

5 References

Alexandersson, J. and N. Reithinger (1995). Designing the Dialogue Component in a Speech Translation System - a Corpus-Based Approach. Corpus-Based Approaches to Dialogue Modelling, Enschede, the Netherlands, University of Twente.

Andernach, T, H. ter Doest, R. op den Akker, J. Schaake, G.F. van der Hoeven, S.P. van de Burgt and A. Nijholt. Language analysis for dialogue management in a theatre information and booking system. Language Engineering 95, AI95, 15th International Conference, Montpellier, June 1995, 351-362.

Aust, H. and M. Oerder (1995). Dialogue Control in Automatic Inquiry Systems. Corpus-Based Approaches to Dialogue Modelling, Enschede, the Netherlands, University of Twente.

Baekgaard, A. (1995). A Platform for Spoken Dialogue Systems. ESCA Workshop on Spoken Dialogue Systems. Vigsø, Denmark.

Bos, D. H. R. (1996). Modelling Dialogues with Finite State Automata. University of Twente. DahlbŠck,

N. and A. Jšnsson (1986). A System for Studying Human-Computer Dialogues in Natural Language. NLPLAB IDA Linkšping University.

Dahlback, N. (1995). Kinds of Agents and Types of Dialogues. Corpus-Based Approaches to Dialogue Modelling, Enschede, the Netherlands, University of Twente.

Doest, H ter , M. Moll, R. Bos, S.P. van de Burgt and A. Nijholt. Language Engineering in Dialogue Systems. Computers in Engineering Symposium. Session on Natural Language in Human-Computer Interfaces. Houston, Texas, February 1996, to appear.

Fraser, N. M. (1995). Messy Data, What Can We Learn From It? Corpus-Based Approaches to Dialogue Modelling, Enschede, the Netherlands, University of Twente.

Morel, M. A. (1989). Computer-Human Communication. The Structure of Multimodal Dialogue Eds. M. M. Taylor, E. NŽel and D. G. Bouwhuis. Amsterdam, North-Holland, Elsevier Science Publishers B.V. 323-330.

Traum, D. R. and E. A. Hinkelman (1992). Conversation acts in task-oriented spoken dialogue. Computational Intelligence 8(3): 575-599.

Zoltan-Ford, E. (1991). How to get people to say and type what computers can understand. Int. J. Man-Machine Studies 34: 527-547.

---

‡ To appear in Proceedings "Natural Language Processing and Industrial Applications", Moncton, New Brunswick, Canada, 1996. 1 a grammar with rules based on the semantic function instead of the syntactic function.