Robust spoken dialogue systems for consumer products: a concrete application

Xavier Pouteau(1), Luis Arévalo(2)
(1): IPO, Center for Research on User-System Interaction, Eindhoven, The Netherlands (pouteau@ipo.tue.nl)
(2): Robert Bosch GmbH, Corporate R&D, Gerlingen, Germany (arevalo@fli.sh.bosch.de)

ABSTRACT

In this paper, we report the significant results of a fully-implemented voice operated dialogue system, and particularly its main component: the Dialogue Manager (DM). Just like for other interfaces, spoken interfaces require a well-conducted design, implying a good analysis of the users' needs throughout the dialogue. The VODIS project[1] has led to the design and development of a spoken interface for the control of car equipment. Due to the workload caused by the task of driving the vehicle, spoken communication provides a potentially safe and efficient mode of operating the car equipment. To achieve this, we present the main characteristics of the *task model* specified during the design stage, and show how its specific features related to the *spoken* communication allowed to implement a robust dialogue.

1. GENERAL REQUIREMENTS

To deal with the specificities of a spoken dialogue in the car, the activities within the project have been focused on four main tasks: a well-performing automatic speech recognition unit, the design of the interface, the integration of all system modules, and the task model, that serves as a backbone to model all relevant components of the interaction in the integrated system. Out of those, we will especially describe the two last ones, closely related to the content of the *Dialogue Manager* [1]. But at first, we provide here a summary of the outcomes of the two first significant activities, taking into account the specificities of the in-car environment [2].

1.1. Signal processing for a well performing ASR

The vocal interface has been designed to operate robustly in an acoustically adverse environment, i.e., it has been taken into account that the speech caught by the far-talk microphone (mounter on the ceiling of the vehicle) is potentially corrupted by other speech or music signals stemming from the car-audio system *and* ambient noise due to the tires, wind, other vehicles, etc. These distortions are tackled by two different signal-processing modules: the former by an *acoustic echo canceller*, the latter by a *noise reduction* scheme. The acoustic echo canceller operates on the signal coming directly from the far-talk microphone and the audio signal played by the loudspeakers. The electro-acoustic path *loudspeaker-room-microphone* is modelled by a 450 tap FIR-filter with coefficients adapted by an NLMS-algorithm enhanced by special measures to ensure fast convergence and stability in a noisy environment [3]. Thus robust voice operation is feasible even if the driver is listening to an audio source of considerable volume-level. This module has been implemented on a DSP-board plugged into the PC that hosts the entire vocal interface.

The noise reduction module has been merged with the feature extractor of the speech recogniser, which is fed with the signal coming from the echo canceller. A *spectral substraction* scheme as discussed in [4] is applied to the MEL spectral coefficients. The noise power in each frequency band is estimated following the principle of the minimum statistics, i.e. observing the minima in a smoothed version of the power spectral density.

1.2. Interface design

The design of the interface has primarily been influenced by the choice of speech as the main mode of human-machine interaction [5]. This choice is based on two motivations:

- Since the driving task puts heavy demands on the users' gestural channel (their hands), using speech to operate the system does not require additional use of that channel.
- Though the number of functionalities available on the interface increases with the sophistication of car equipment components (the complete system used in VODIS offers more than 80 functionalities), the space available on the dashboard is generally very limited. So opting for a tactile interface would enforce to design a dialogue implying a large number of sub-menus, as well as a high constraint on the driver's visual channel.

To limit the constraints on the user's visual channel, another important point is the amount of information to provide to the user, and the form to convey it. In the remainder of the paper, we will refer to conveyed information as*feedback*, although the definition of feedback itself is more specific than the global notion of providing information. As always, the dilemma between a spoken feedback, a visual feedback or a combined one had to be solved. On the one hand, spoken messages, via text-to-tpeech synthesis (TTS), are actually perceived by users without distracting their visual attention.

But on the other hand, TTS messages are transient, so feedback is only accessible at the time the message is spoken. Furthermore, spoken messages are undoubtedly intrusive, while visual feedback is only accessed depending on users' decision (when they actually *look at* the screen).

To provide the user with the "right" feedback, every dialogue situation has been carefully studied, so as to define:

1. For more information, please refer to http://werner.ira.uka.de/VODIS, and to [6]

- the amount of information required. In VODIS, every feedback has a “sort” and a “long” version, so as to give minimal feedback to experienced users on the one hand, while novice users are given more informative feedback. Furthermore, as far as possible, the wording employed by the users (the vocabulary of the commands) is also used in the feedback, as to help the user learning faster the command and control language of the application.

- the form used to convey feedback (TTS, visual display of a combination of both).

Finally, the vocabulary itself has been defined in close collaboration with car manufacturers and ergonomists of the application area, also using the results of user interviews.

2. PRESENTATION OF THE SYSTEM

2.1. Consistency of the System

In the general architecture, the System Manager (SyM) serves as a software interface to the Dialogue Manager. In this respect, it allows the control of the components by DM, in order to switch the set the components to a new state (e.g., selecting CD player), as well as to read information about available data in the system (e.g., description of CDs of the CD player).

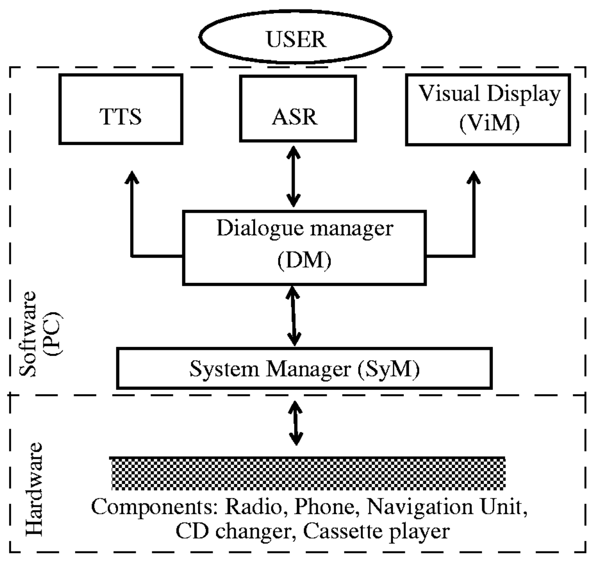

Figure 1: architecture of the system.

Furthermore, SyM also sends messages to DM to indicate changes of state of the system: to keep the example of the CD player, new CDs can be inserted. Thus, DM can detect that a new list of CDs is expected, but also, during the insertion of a CD, the CD player itself cannot be selected as current audio source.

This communication between DM and SyM plays a crucial role in the complete system, since it ensures the consistency of the modules during the dialogue with the user. In fact, it is necessary to take into account three aspects of each component:

Operability of the components: transparency

The hardware components can be represented as finite-state automata. As such, only a subset of all available operations on a component is allowed at any time. This description is used in DM to keep the characteristics of the components hidden to the users, so they can operate the system in a transparent way. Let us illustrate this by an example.

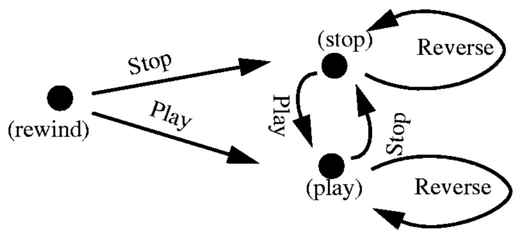

Figure 2: partial description of an automaton (example).

The cassette player does not allow to “reverse” the side of the tape when rewinding the tape. If this would be held also in the dialogue, then the user would have to perform three operations to reverse the tape: at first stop rewinding, then reverse the side, and finally play (figure 2).

Instead, the command “reverse” operates in this case as a macro, for the sequence of three commands above. This approach has both advantages of being able to deal with different kinds of hardware characteristics, and also to modify the characteristics of the functionalities at a software level. In our example, this allows to use a unique command for the transition between the states “rewind” and “play”. Furthermore, this kind of improvements of the dialogue also dramatically decreases the workload drivers have to spend on the control of the system, thus allowing a better attention to the traffic situation.

Operability of the components: robustness

Of course, the communication between DM and SyM also serves the purpose of ensuring that no “illegal” action is performed. This is a generalization of the case above, where no path can be found to reach the desired state. For example, when there is no cassette in the cassette player, no operation but the insertion of a cassette is possible, and this allowed operation cannot be achieved by DM. This aspect of the communication between DM and SyM thus avoids severe problems, by preventing any erroneous operation in the hardware components to be attempted. It is shown in section 3 how this has influence on the dialogue.

Components data: dialogue consistency

The last role of SyM for the application is to provide information on the state of data related to the components. The components of the system used in the project offer several possibilities, such as giving a name to a CD, reading the name.of a Radio Station (RDS information), etc., as indicated on the table below. The names used as parameters in the commands are stored in the Lexicon (see section 3), and dowloaded to the ASR unit (which makes them available for further recognition via a grapheme-to-phoneme conversion).

Component	Related data
Radio	List of Radio Station Names available
CD changer	Place of present CDs (out of 10 places in CD changer) Names of present CDs
Telephone	List of Phonebook Addressees
Navigation unit	List of pre-stored destinations

3. THE TASK MODEL: THE SUPPORT FOR A ROBUST DIALOGUE

Most functions available at the user interface can be expressed by a single command: selection of a component (“Radio”, “Cassette”, “Telephone”), or specification of a parameter of those components (“Call Luis” to dial the phone number of an addressee in the phonebook, “CD Bach” to play corresponding CD, “Track 4” to select a given track on the current CD, etc.). In those cases, the dialogue consists of just one command (and its related feedback).

In some cases however, the function selected by the user (and its corresponding *task*) requires several parameters to be indicated. In those situations, the dialogue is conducted by DM at each step to control both the results of the ASR unit and the consistency of data entered by the user.

3.1. Example: Entering A Destination

The system comprises a navigation unit capable of performing route guidance by an *incremental* route description (in the form of on-time instructions: “turn right in 150 meters). To specify the destination they want to be guided to, users can either choose out of a list of pre-stored destination, or compose a new destination. A typical example of the profile of a destination is a city name and a street within this city. Making use of GPS information, the navigation unit can identify the current position of the car and calculate the route to the destination. In the transcription below, *long* feedbacks are used (see section 2)..

User	System	Comments
“Enter destination”	Please indicate the part of the destination to specify	The screen displays the options: city, airport, street, highway exit
“City”	Please say the name of the city	DM activates the ASR on city name list
“Berlin”	“Berlin”	The screen displays option yes/no.
“Yes”	Please indicate the part of the destination to specify	The screen displays “City: Berlin” and the options of first screen
“Street”	Please say the name of the street	DM activates ASR on the list of street names of Berlin
“Branden burg street”	“Waldenburg street”	DM prompts for the candidates of the N-best list returned by ASR with a good score User’s speech signal is recorded
“No”	“Neubrandenburg street”	next possible candidate in the N-best list
“No”	“Please spell the beginning of the name of the street”	No other plausible candidate. DM prompts to spell to gather a sublist of all the streets.
“B R A N”		DM activates ASR to perform recognition of recorded signal on the list (result of spelling)
	“Brandenburg street”	New N-best list is processed
“Yes”	“Please indicate the part of the destination to specify, or say over”	The screen displays the options, in addition to “over”
“over”	...	

3.2. Presentation Of TheTask Model

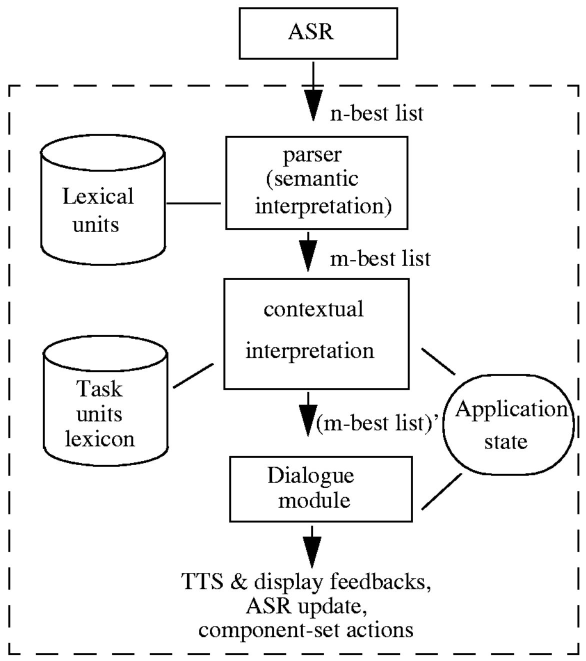
ASR
n-best list
Lexical units
parser (semantic interpretation)
m-best list
Task units lexicon
contextual interpretation
(m-best list)’
Application state
Dialogue module
TTS & display feedbacks, ASR update, component-set actions

Figure 3: content of the Dialogue manager

The dialogue management is implemented in lexical units available in a lexicon of the Dialogue Manager, called task units. Those task units constitute together the Task Model, and correspond with the functionalities available at the user interface. Every task unit comprises the following aspects:

Preconditions

The preconditions of a task test whether the system state allows to perform the task chosen by the user (availability of components, see also example of figure 2). As an extension, the concept of preconditions has been enlarged to include *prerequisites* in the dialogue.

ASR unit operations

When a task is performed, the system is set in a new state, and the dialogue itself reflects the characteristics of this new state by selecting in the ASR grammar the utterances allowed by users during their next turn.

Dialogue strategies

As illustrated in the dialogue sample in part 3.1., the dialogue between the user and the system cannot only rely on a simple scheme *user command/system reaction*. This avoids error loops (users making several similar attempts, all leading to recognition errors), and dramatically improves the efficiency of the dialogue. Dialogue strategies are deployed in dialogue scripts where DM takes the initiative by prompting users's next input.

Feedbacks

Feedbacks have been defined as described in part 1.2. They take place after and before each turn in the dialogue (which does not mean that each turn is associated with a message, since feedbacks can also be empty).

Post conditions

Post conditions are meant for testing that the system state is set accordingly after the users have performed a task. Non statisfaction of post-conditions can be cause by a problem with one of the components, and is notified to the user via a feedback.

3.3. Report On The Implementation

The system was initially meant to be operated in French and German, and both versions are fully-implemented. Because of the general design approach based on lexicons, the system is actually independent of the language, and can easily generate a version in another language, just requiring the time necessary to edit vocabulary and feedback messages files, for a global effort of only a few days.

The implementation of DM has been achieved following to a real-time oriented specification of the system, so as to be able to handle the multiple-write and read accesses of the modules on some sets of data. Furthermore, the performances of the system are real-time or near real-time during the dialogue: the longest delays do not exceed a few seconds in the case of very long lists of street names (at times more than 12,000 or 15,000 entries) for the biggest cities. For all other cases, the vocabulary is more limited, and the reaction delay of the system from the end-of-speech time, including speech recognition, and excluding operations that cannot be compressed[2], is inferior to one second.

4. CONCLUSIONS

This project has shown the technical feasibility of voice-controlled interfaces for command and control applications. User evaluations are currently being carried out, and will allow in the coming monthes to draw conclusions on the expected benefits of spoken interfaces in terms of accessilility, acceptability, workload and efficiency of such interfaces. We have shown in this project that the global design of a spoken interface basically consisted of an superset of the design of "classical" tactile (gestural/visual) interfaces. In this perspective, the design must at first concentrate on the analysis of the user's activity, and on the basis of the practical charateristics of voice technologies, extend the design towards the integration of speech in the relevant models in an adequate manner. Then, the complete system must include mature speech technology, since a sifficiant performance in speech recognition is a mandatory pre-requisite for a spoken interface. Finally, the integration of the complete system is the last important phase that allows to achive a robust prototype. At this stage in the development of voice-operated systems, evaluation of the system serves as the last important step to validate the hypotheses used in the design of the system, and to improve the prototype up to a consumer-usable system.

5. REFERENCES

1. Pouteau, X., Krahmer, E., Landsbergen, J., "Robust Spoken Dialogue Management for Driver Information Systems", *Proceedings of Eurospeech 97*, 2207-2210, 1997.
2. Leiser, R., "Driver-vehicle interface: dialogue design for voice input", in *Driving future vehicles*, A. Parkes, S. Frazen (eds) Taylor & Francis, 275-294, 1993.
3. Antweiler, C., "Annual Report on Acoustic Echo Control", *VODIS Deliverables D007-1/2*, Aachen, 1998.
4. Cappè, O., "Elimination of Musical Noise Phenomenon with the Ephraim and Mallah Noise Suppressor", *IEEE Trans. on Speech and Audio Processing*, 2, 345-349, 1994.
5. Lea, W., "Developing usable voice interfaces", *Journal of the American Voice I/O Society*, 16, 1994.
6. Geutner, P. Denecke M., Meier U., Westphal, M., Waibel, A., "Conversational Speech Systems for on-board Car Navigation and Assistance", *Proceedings of ICSLP 98.*

Ackowledgements

This work has been partly funded by the European Commission in the framework of the Language Engineering/Telematics Applications Program, project LE-2177 (VODIS).

2. For example, TTS messages, or the reaction delay due to mechanical components, like when the CD changer inserts a new CD in the CD player.