Findings with the Design of a Command-Based Speech Interface for a Voice Mail System

Stephan Gamm, Reinhold Haeb-Umbach, Detlev Langmann
Philips Research Labs
P.O Box 1980
52021 Aachen, Germany
Tel/Fax.: +49 241 6003 500/518, E-mail: gamm@pfa.research.philips.com

ABSTRACT

This paper tells the story of the design of a command-based speech interface for a voice mail system. Speech recognition was integrated in the voice mail system in order to allow the remote interrogation of messages in a speech-only dialogue. Our design goal was that consumers would perceive voice control as a clear benefit versus touch-tone control. It is shown how the speech interface was designed in a top-down approach. We started with a concept development and tested it by means of a Wizard-of-Oz simulation. After refining the concept in parallel design, the design was implemented in a high-fidelity prototype. By means of qualitative user testing it was improved in three iteration steps. We verified the achievement of our design goal with tests in two countries.

INTRODUCTION

Telephone answering machines and network-based voice mail systems for the public telephone network have found widespread acceptance over the last years and have become a real consumer product [1]. Messages left by callers can be interrogated from any point in the world over the telephone line. From afar the systems are mostly controlled via touch-tones, the DTMF signals. For each function there is a number code which corresponds to a certain DTMF signal. However, DTMF control is burdened with two shortcomings:

- Users need additional beepers
  Not all telephones can generate the DTMF signals. In many countries DTMF penetration is below 60% [2]. Therefore an additional device is needed to generate these signals. This results in additional costs, and the beeper must be carried along all the time.
- Users need memory cards
  All the number codes for the functions are difficult to remember. Hence memory cards are provided on which the number codes pertaining to the functions are listed.

Voice control overcomes these shortcomings: The user's voice is always "at hand" and the mental load for remembering command words is lower than for number codes.

The paper is organized as follows: First we describe the speech recognition technology underlying the system. Then we show how the speech interface was developed in a user-centered design process. We present the conceptual design and how it was evaluated by means of a low-fidelity prototype. Then we show how we came to the initial design and how this was improved in an iterative process. At last we describe the final evaluation of the speech interface.

THE UNDERLYING TECHNOLOGY

Users tend to communicate with machines like with human beings. They do not stick to the accepted command words and embed them in longer phrases. In the speech data collection for the voice mail system for example, 6% of the prompted command words were not spoken at all and 10% extraneous utterances were recorded. Therefore a robust rejection of out-of-vocabulary words is required besides the speaker-independent recognition of the command words.

The underlying wordspotting task is performed using the Philips continous-speech recognition framework [3]. It is based on the statistical modeling of spoken speech by means of left-to-right Hidden Markov Models (HMM) with continuous mixture densities. Keywords are represented by whole-word HMMs, and the rejection model consists of a network of parallel filler phoneme HMMs and a background HMM. The models have been trained according to the maximum likelihood principle by an iterative estimation-maximization procedure.

The speech recognition is performed by Viterbi decoding and time-synchronous one-pass beam search. A finite state grammar determines all allowed keyword sequences. In addition, each grammar node contains a self-loop assigned to rejection models as permanent alternative to the keywords. Separate global penalties for the keywords and the rejection models can be choosen to adjust the tradeoff between the probability of correct keyword detection and the false-alarm rate.

CONCEPT DEVELOPMENT

The starting point for the design process was a functional specification. The remote interrogation should comprise the following functions.

- Navigate to next / previous message
- Replay single / all message(s)
- Delete single / all message(s)
- Record a new greeting
- Deactivate the answering machine

The outcome of the concept development was an outline of the man-machine interaction on a rather abstract level. Its objective was to define the structure of the speech dialogue. At this

0-7803-3238-5/96$5.00©1996

stage it was not intended to cover all possible paths of the dialogue. The concept was formally specified by means of flowcharts.
We considered compatibility as a very important design principle [4], especially when introducing a new interaction technology such as speech recognition. In answering machines the remote interrogation with touch-tones is a common feature and many people are already used to it. These people should be able to transfer the knowledge they have already gained. Therefore we adopted the concept used for DTMF control also for voice control. In this concept the dialogue is divided into two parts (see fig. 1).

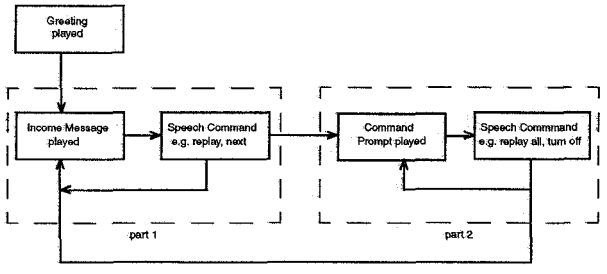

Figure 1: Outline of man-machine interaction

The first part is when messages are being played. Here the user can activate functions that affect a single message. The user can replay or delete the message that she/he just heard, or she/he can navigate to the next or previous message. After having heard all messages, the dialogue enters the second part. Here the user can activate global functions that affect all messages or the greeting. She/he can delete all messages at once, start replaying all messages, record a new greeting or deactivate the answering machine.
This concept is well-established for touch-tone control, but that does not necessarily mean that it is successful for voice control as well. Yankelovich et al. showed that graphical user interface conventions for example could not successfully be transferred to a speech-only environment [5].

In order to avoid to go into the wrong direction at this early stage of the design process, we evaluated the concept in a small-scale user involvement test. For these tests we used a Wizard-of-Oz simulation [6]. Here the speech recognition and the system control were both performed by a human operator. So, the simulation itself did nothing more but generate the speech output on certain button clicks. In a control panel the human operator could trigger the playback of different messages and system announcements depending on what the user said

TAM:	This is the first message: 'Hi, Steven calling ...'
User:	Next message.
TAM:	All messages have been played. Do you want to execute further functions?
User:	Yes.
TAM:	If you want to edit the messages please say 'messages'. For editing the announcement please say 'announcement' and if you wish to turn off the answering machine please say 'turn off'.
User:	Turn off.

The purely qualitative user involvement test was conducted with eight test persons and took 4 days. It turned out that the concept was generally accepted. Some people were uncertain when to speak, but the structure of the dialogue was clear to most of them. All people who were used to answering machines recognized the structure.

INITIAL DESIGN

Parallel design

The initial version of the speech interface was created in parallel design [7]. Independently from each other the concept was refined at two different locations: at the research lab and at the development site. The resulting versions differed mainly in two points, namely whether the user may be enabled to skip the first part, and whether the user should be guided by acoustic menus in the second part.
In version one the user was forced to listen to the received messages before entering the second part of the dialogue. After the user identification the system began with the announcement: *"There are messages received. Please listen."* Thereafter it immediately started playing the received messages. In version two however the system asked: *"Do you want to hear the received messages?"* Answering with *"no"* would have skipped the first part of the dialogue.
The other basic difference concerned the user guidance in the second part. In version one the second part began with the general prompt *"Please give an instruction."* In version two, however, the user was guided by acoustic menus, similar to interactive voice response (IVR) systems. The system asked: *"Do you want to execute further functions?"* Responding with *"yes"* would have started the menu selection: *"If you want to edit the messages please say 'messages'. For editing the announcement please say 'announcement' and if you wish to turn off the answering machine please say 'turn off'."* After selecting a group of functions, e.g. messages, the system would have offered the available functions: *"If you want to hear the messages please say 'replay', and if you want to delete all messages please say 'delete'."*

The initial design

In a meeting in which all designers, the technical developers and the responsible product managers participated, both versions were merged to an initial design. For safety reasons it was agreed that it should not be possible to skip the first part. People should be forced to listen to their messages first, before being enabled to delete them all at once. The question about user guidance was basically a trade-off between the efficiency and the ease-of-use of the system. Both criteria were equally important in this application: The remote interrogation should be fast, because calling long-distance is expensive; but it should also be intuitive in use, because there would be no manual at hand. Version one was efficient, and version two was easy-to-use, but none of them was both. Therefore we decided to combine both versions by implementing them as two modes, the *command mode* and the *menu mode*. The appropriate mode is automatically determined by the system. The default mode is the command mode. Here the system just prompts for an instruction and the user speaks a command word. If the user reacts in an inappropriate way, e.g. by using invalid command words or by not reacting at all, the system goes into the menu mode and guides the user with menus. By

speaking *"help"* the system can also be explicitly switched to this mode. After a successful activation of a function the system falls back onto the command mode.
In the menu mode as described above the user had to select a group of functions first, e.g. editing the messages, and then she/he could select a particular function, e.g. deleting all messages. For efficiency reasons we decided to give up this hierarchy and we offered the functions directly: *"If you wish to hear all messages again, please say 'replay', to delete all messages say 'delete' and if you want to execute other functions please say 'other'."* Due to the number of functions, only the most frequent ones could be directly offered. The less frequent ones were offered in another menu that was triggered with the command word *"other"*.
The dialogue was formally specified by means of a finite state grammar . The whole interaction was modelled as a state diagram with 62 different states and 965 state transitions among them.

ITERATIVE IMPROVEMENT

The Initial Prototype

In order to test the initial design with users we built a high-fidelity prototype according to the specification. The prototype was in fact a pure software simulation but that was hidden to the user. She/he interacted with it via a normal telephone set. For her/him there was no difference at all to a real product. This realistic impression certainly made our test results more reliable.
To be more efficient in the following iteration process we also developed a dialogue editor with which we could easily change the system (see fig. 2).

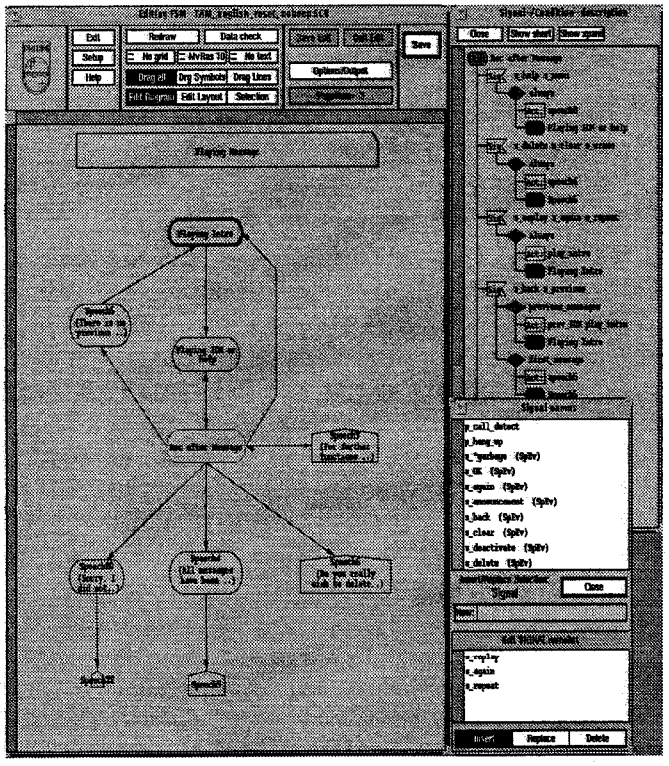

Figure 2: Dialogue editor

With the prototype we conducted purely qualitative tests. We gave the test users a set of representative tasks, observed them during the execution and interviewed them afterwards. Moreover we asked for feedback from everyone who saw the system in a demonstration. In each iteration step we wanted to identify one, most urgent problem.

First Iteration - Intuitive command words

In the tests it turned out that some of the command words we chose for the vocabulary were actually not used. In order to identify a more intuitive set of command words, a synonym test had been conducted. In this test the menu mode was turned off so that the users had no clue of which the accepted command words were. Having given them a certain task, it was observed which command words the users would intuitively use. It turned out that some command words, such as 'delete', were intuitively clear, whereas for some functions, such as changing the greeting, no common command word could be identified. For those functions several synonyms were included in the system's vocabulary. As a result of this test the vocabulary was nearly doubled.

Command Word	Synonyms
help	menu
new announcement	new, announcement, greeting
no	not
previous	back
replay	repeat, again
turn off	switch off, deactivate
yes	OK, sure, of course

Table 1: Synonyms for command words

Second Iteration - Efficiency

After changing the vocabulary we started the second iteration step. In further tests we identified as the next most urgent problem the subjective impression of the efficiency. The system was generally perceived as being too slow. This perception might have resulted from both directions of man-machine communication: Recognizing the speech input might have taken too long and the speech output might have been too sluggish. Due to the simulation environment we could not further shorten the response times, so we could only improve the efficiency of the speech output.

- Speaking rate of system announcements
  Taken into account the transmission via a long-distance telephone line, the announcements were pronounced very clearly, with a speaking rate even slower than in the news. But this was perceived as being too sluggish. We thereupon re-recorded the system announcements and increased the speaking rate by 35%. This led to a satisfying perception by the users. All system announcements had been spoken by a professional, female speaker.

- Length and content of menus
  It was not possible to interrupt the system, so the user had to wait for the end of an announcements. Long menus were perceived as sluggish, especially when more information was given than necessary for a particular user. It often happened that users just forgot the valid command words, but they knew what they wanted. Therefore we split the help menus into two levels. In the first level the user was just reminded of the command words: *"The accepted speech commands are 'replay', 'delete', 'new announcement' and 'turn off'"*. Only if then again the user did not respond with a valid keyword, the system played the second level of the menu. Here the functions were explained and the speech commands as well as the DTMF commands were mentioned:*"If you wish to hear all messages again, please say 'replay' or press <6>, to delete all messages please say 'delete' or press <7> and if you want to execute other functions please say 'other' or press <5>."*. The DTMF commands were introduced in order to offer a fallback mechanism in case of repeated misrecognitions [8].

Third Iteration - When to speak

After each system announcement a short beep prompted the user to speak. She/he had then two seconds of time to respond. In further tests it turned out that this fixed time window seemed to be unacceptable. It was observed that users either tried to barge into the announcement or they waited too long and the system did not recognize the command any more.

- Begin of time window
  Because implementing a barge-in facility is a major effort, we decided to first observe the user's behaviour in more depth. It turned out that only very few people consciously tried to interrupt the system in the midst of an announcement. What mostly happened was that people impatiently waited for the end of an announcement. They started speaking immediately after the last syllable of the announcement. Thus they overspoke the following beep and unconsciously tried to barge into the system. This happened even if people knew that a beep after the announcement would explicitly prompt them to speak. From that we derived that the announcement itself is prompt enough, and we left out the beep. Observations revealed that it was not missed and less people barged into the system.
- End of time window
  In order to give also the slower user a chance to respond, the time limit was extended to five seconds. The end was made flexible in that sense that the system reacts as soon as a command word has been recognized. With this flexible end it was possible to satisfy the slow user as well as the fast one.

VERIFICATION OF DESIGN GOAL

Our design goal was that consumers would perceive the voice control as a clear benefit versus DTMF control. For verifying the achievement of this goal an acceptance test in form of focus groups was conducted. There were three focus groups each in two countries, France and Germany. People in the same focus group had similar backgrounds in using telecommunication terminals. There was one group of 'basic' users, who owned just a plain old telephone. The second group were users of a fax or an answering machine either at home or at the office, and the third group were mobile telephone users. Each group was composed of men and women, aged between 25 and 50. In total 48 persons participated in the tests. After a general discussion about the technology, people were given a demonstration of the system and everybody had the chance to try it out. Feedback was collected first in a questionnaire and then in a group discussion.
The result was that users saw a clear advantage compared to commercially available systems with DTMF control. The preference was clearer in Germany than in France. Being independent of the availability of DTMF and thus getting rid of an additional beeper was the basic reason for this preference. This motivation explains the clearer preference in Germany, where DTMF penetration is much lower. The fact that speech commands are easier to remember than number codes was not the decisive reason. People said they can remember the codes for the very few functions they actually use. Nevertheless the guidance by acoustic menus was appreciated very much. People liked the conversational behaviour of the answering machine, they did not feel alienated too much when talking to a machine. As a conclusion we saw our design goal as achieved and finished the interface design process.

REFERENCES

[1] The Yankee Group(1990). Voice Messaging: The Roll-out Begins, Yankeevision, July 1990

[2] Bennett, R.W., Syrdal, A.K., Halpern, E.S. Issues in designing public telecommunicarions services using ASR. Proceedings of Speech Technology '92: Voice Systems Worldwide, 1992, 222-229

[3] Ney, H., Steinbiss, V., Aubert, X., Haeb-Umbach, R.: Progress in Large Vocabulary, Continous Speech recognition, In: Niemann, H., de Mori, R., Hanrieder, G.(Eds.): Progress and Prospects of Speech Research and Technology, 1994

[4] The Guidance Project (RACE 1067), Usability engineering methods for IBC services, Publications for RACE, 1992

[5] Yankelovich, N., Levow, G.A., Marx, M. Designing SpeechActs: Issues in speech user interfaces, Proceedings CHI '95, Denver, 1995

[6] Gould, J.D., Conti, J., Hovanyecz, T. Composing letters with a simulated listening typewriter. Communication of the ACM 26, 4(April) 1983, 295-308

[7] Nielsen, J., Desrvire, H., Kerr, H., Rosenberg, D., Salomon, G., Molich, R., Stewart, T. Comparative design review: An excercise in parallel design. Proceedings INTERCHI '93, Amsterdam, 1993, 414-417

[8] Falck, T., Gamm, S., Kerner, A. Multimodal dialogues make feature phones easier to use. Proceedings of Applications of Speech Technology, Lautrach, September 1993, 125-128