E-MAIL GOES MOBILE: THE DESIGN AND IMPLEMENTATION OF A SPOKEN LANGUAGE INTERFACE TO E-MAIL

Daniela Oria, Esa Koskinen

Speech and Audio Systems Laboratory
Nokia Research Center, Helsinki, Finland
daniela.oria@nokia.com, esa.koskinen@nokia.com

ABSTRACT

EVOS is a multilingual speech-enabled e-mail client that allows users to access their e-mails by using their voice. In this article we describe the implementation of the spoken language interface to e-mail and we motivate our choice to adopt a system-driven rather than a mixed-initiative interaction style. We also present the results of two usability tests and we compare them to those obtained from the evaluation of a DTMF interface for the same system.

1. INTRODUCTION

In recent years the utility and appropriateness of spoken language interfaces (SLIs) have been heavily debated. Acquiring and presenting data making exclusive use of the auditory channel, without the graphical support of a screen, is very challenging: the information that would otherwise be displayed on a screen has to be "translated" in an audible format, which must be easy for the user to follow and to remember. The spoken interface must be "transparent" and clear, so that the user is not lost and always knows at which stage of the interaction he is and how he can proceed to accomplish his task. Audio-only presentation of data is slower in comparison to visual presentation. In addition to this, speech input has to fight with the limitations of automatic speech recognition (ASR) technology.

Nevertheless, our suggestion is that SLIs are not in competition with graphical user interfaces (GUIs), but rather offer an alternative to them, and in certain situations they might prove to be more efficient and convenient than their graphical counterparts. As an example, let us take the case of a SLI to e-mail. The need to access e-mail and other work-related information anytime and anywhere is constantly increasing and the mobile phone (and thus speech) seems to be one of the ideal ways to quickly connect to such data. Our work is located in the framework of a large number of studies aimed at finding the most appropriate way to interact with an e-mail application over the phone [1] [2] [3] [4] [5].

EVOS (E-mail Voice Service) has been developed in the scope of a larger concept that allows "all-round mobile connectivity", whose aim is to realize the concept of "mobile office" by allowing people to access their personal information (e-mail, calendar and address book) anytime, anywhere through different interaction modes (SMS, WAP, web, and voice).

This paper presents the design and implementation of the multilingual speech-enabled e-mail client and it discusses the results of two usability studies aimed at testing and evaluating our interface and at comparing it to a DTMF interface for the same service. Our results show that users get familiar with the spoken language interface quickly and that they are willing to use the system also in the future.

2. SYSTEM ARCHITECTURE

Figure 1 presents the architecture of EVOS. Call control (answering and terminating user's calls) is handled by IBM's Direct Talk telephony platform, which runs on an ordinary PC server that is connected through an ISDN primary rate line to a local telephone switch.

![System Architecture Diagram](image_url)
to the user's inputs. In the case of incomplete or missing information (resulting from too low recognition confidence or time-outs), it can prompt the user back and ask for more input. Once it has enough parameters, the dialog manager connects to the back-end and fetches the requested information.

The speech recognition module used in our application is IBM's Via Voice Telephony, a speaker-independent large vocabulary continuous speech recognizer for American English. The TTS engines currently being used are Scansoft's (formerly Lernout & Houbre) American English and IBM's Finnish.

2.1. Language identification

A considerable portion of the e-mail traffic in our research laboratories consists of messages where the text is either written in languages other than English or different languages are mixed within the same message. This is often the case of Finnish e-mail messages, where the text is mainly in Finnish, but English words or whole sentences in English (maybe quoted from other messages) are inserted in the Finnish text. In such a situation, it is quite obvious that offering only an English TTS for reading out the e-mails would be a considerable limitation. Quite often users would end up having, for example, Finnish text read with the English synthesizer, which of course would greatly hinder the usability of the application.

For this reason we introduced a statistical, tri-gram based language identification module. Before being passed to the TTS engine, the back-end output (i.e. the list of messages in the user's inbox or the e-mail full text) is passed through the statistical language detection module. This checks the text sentence by sentence, identifies the language each sentence is written in, and accordingly selects the correct synthesizer. Since language identification is done sentence by sentence, it is possible to switch from one language to another while reading the text of the same e-mail: if a message contains parts in different languages, each portion will be read using the appropriate synthesizer.

In the first prototype of our application (the one used in the usability tests presented in this paper), language identification was supported only for English and Finnish and proved to be working well.

The sentence based language detection approach, however, still doesn't solve the problem of having, for instance, English words inserted in Finnish sentences or vice-versa. Since the chosen TTS corresponds to what is identified as the "predominant language" of the sentence, Finnish words in a predominantly English sentence will be read with the English synthesizer, which might hinder the comprehension (especially in the case of proper names or addresses). In order to overcome this problem, language detection should be done word by word and different TTS engines should be used within the same sentence. However, word-based language identification is less reliable than the sentence-based one, and frequently switching synthesizer might also be disturbing for the user. This still remains a problematic issue in our application and it definitely needs further investigation.

2.2. E-mail pre-processor

After the correct language has been identified and prior to being passed to the TTS engine, the output text is passed through an e-mail pre-processor module, which takes care of formatting the back-end output in a "speakable format". E-mail text is particularly irregular and contains a lot of non-linguistic structural elements, which become redundant when they have to be presented orally and therefore need to be handled appropriately.

Here are a few examples of problematic issues that are handled in our pre-processor:

- **e-mail header**: message sender and subject are presented to the user, date and time when the message was received and other details sent by the mail server are discarded
- **non-linguistic structures, such as "ASCII graphics"** (sequences of asterisks or hyphens, separators or similar "art work") are discarded
- **tables and inserted objects** are not handled, the user is just informed that the message contains an embedded object or table
- **abbreviations and emoticons** are expanded
- **attached files**: the e-mail reader mentions only the name and file type of the attachment. Audio files are played to the user.

Each language has its own pre-processor, which is selected according to the result of the language identification module.

3. DESIGN OF THE SPOKEN LANGUAGE INTERFACE

EVOS is operated by using voice commands. The interaction is system driven: the system guides the users step-by-step through the whole interaction, telling them what kind of functionalities are available at each stage and suggesting which voice commands should be used in order to activate the desired function.

We chose this system-driven approach after doing some experiments with a very open mixed-initiative interface (similar experiments were reported in [2], [3] and [4]). In the mixed-initiative version, our system did not guide the users at all. It did not give them any hints about the supported functionalities; if the users were lost, they could find out what they could do by using the "help" function. As for the supported commands, our aim was to allow users to operate the application by using natural language expressions. We did not prompt them to use predefined commands from a limited set, but rather compiled a very big grammar that also covered complex natural language expressions. Figure 2 shows an example dialog with the mixed-initiative interface:
Because of the complexity of the grammar, the speech recognizer also did not perform well enough. Recognition accuracy was worsened even more by the fact that users were calling from mobile phones (the quality of the speech signal decreases when the speech input is coming from a mobile device) and often from noisy environments, such as from a car while driving. The high amount of misrecognitions made users frustrated and confused. Rejections and misrecognitions on expressions that were actually included in the grammar gave them the impression that the application didn't work because they were not using the right commands. At the same time they did not know what else to say to operate the application.

As a result of this experience, we redesigned the interface and adopted a system-driven approach. Figure 3 shows an example dialog with the system-driven interface:

S: Welcome to e-mail service. Hello John Smith! You have 5 new messages. Message 1 from Susan Johnson. Subject: Project meeting. Say “read message” to listen to this message. “next header” to listen to the next header or “goodbye” to exit.
U: Read message
S: Fetching message number 1, please wait. Reading message number one...[Reads message]. Say “repeat message” to listen to this message again, “send reply” to reply to this message, “send to my mobile” to receive this message to your mobile phone...

Figure 3: System-driven interface

This kind of interaction does not leave much space for user's initiative, but it's very efficient in the case of novice users. Since system prompts are very guiding, users can direct the application without having to read manuals or memorize commands. This significantly reduces the strain on users and also prevents them from being lost in the dialog, as they are always reminded of how they can proceed.

Prompting users from a limited set of commands also increases recognition accuracy. Based on what we learnt from our experience with the mixed-initiative interface, supporting a big and comprehensive vocabulary does not seem to be crucial for the e-mail reading domain: users seem anyway to be interacting with the application using a quite restricted vocabulary and uttering short commands rather than more complex, natural-language-like expressions.

One of the reasons that justify the existence and usage of a speech-enabled e-mail reading application is that it allows users to connect to their inbox and take care of their e-mails while on the move (or anyway out of office). It is therefore not very likely that users will call from a silent, office-like environment. When using a mobile phone, one of the aims is also to be able to call the application in a hands-free environment, such as when driving a car and using a headset.

In such situations, improving recognition accuracy and system performance so that users will be able to perform the desired task in noisy environments is much more important than offering a more natural interaction style.

4. COMPARISON OF THE VOICE INTERFACE WITH A DTMF INTERFACE

In our usability tests we compared the system-driven voice interface with a DTMF interface developed for the same system. In both applications the call flow, the functionalities supported and the structure of the interaction are the same. The only difference is in the way users can control the system, i.e. with spoken commands or by pressing keys on the telephone keypad, respectively.

Two kinds of usability tests were conducted. In the first one, a group of novice users (9 users) was asked to use both systems for two weeks and evaluate them. They were not given any specific task to carry out, but rather were instructed to use the applications as they would do in real-life situations. They didn't receive any manual explaining how to use the system, so they had to learn to do so by listening to the system prompts.

The second evaluation was conducted on a group of users (9 users) that had been actively using both systems for a couple of months.

In both cases, users' responses were grouped in order to evaluate the user-friendliness of the system, its reliability, efficiency and ease-of-use, the naturalness of the interaction, the cognitive load on the user, the users' attitude towards the system and their perceived success rate. All aspects received scores on a scale from 1 to 5.

In the case of both user groups, the application with DTMF commands received slightly better scores. The first user group perceived it as more efficient, easier to use, and requiring less cognitive effort. This was probably influenced by the fact that the usage of DTMF commands has a very low error rate in comparison to speech recognition. Moreover, users maybe felt stressed out by having to interact with the system in a foreign language (most test users were Finnish native speakers, whereas the system interface is in English).

In the second test group we noticed that keen users of the speech-enabled application were usually quite enthusiastic...
about it, and they appreciated its ease-of-use and the naturalness of the interaction. The interface proved to be clear, so that users quickly learnt to use the system and easily adopted this new interaction mode (see Figure 4 for an overall view of the evaluation results).

Two aspects were particularly encouraging. First of all, long-time users had a positive attitude towards the speech-controlled application (the average score in response to the question "Would you be willing to use this application in the future?" was 4.25). The other very positive result was in response to the question that asked users how fast they learnt to use the application (the average score was 4.75). This shows that it might take some time before people get used to interacting with a system using speech, but then they start to appreciate the benefits of this kind of interaction, that in certain situations overcome the shortcomings of the technology.

In our results we noticed another very interesting tendency. Users that had used DTMF-driven applications prior to taking part in our tests but had no experience with speech-controlled application tended to prefer the DTMF interface to the speech one. On the other hand, people that were familiar neither with DTMF nor with speech interfaces tended to prefer the speech-enabled application. Even though our data is not enough to claim that this is a statistically relevant pattern, this behavior might show two substantial tendencies:

- users that are familiar with DTMF interfaces tend to stick to the interaction mode that they have already learnt to use, and they are therefore more "resistant" to adopt a new interaction style
- novice users that have experience neither with DTMF nor with voice interfaces (and therefore are not biased by previous experiences) usually prefer the speech mode since it is more intuitive, quick to learn and easier to use.

5. FUTURE WORK

Future research will include finding a way to allow users to retrieve relevant information quicker than in the current version (without the need to "scan" the entire inbox to find relevant e-mails) and developing an adaptable interface for expert users, which will make the interaction less system-guided and more open to user initiative. In doing this, we will have to keep in mind the lessons learnt while testing the mixed-initiative interface and find the correct balance between a too open and a too system-directed interface.

In future prototypes, the language identification module will be improved to support more languages (German, French, Spanish and Italian). We also plan to make the e-mail pre-processor more sophisticated, so that for example it would be able to handle language or sub-domain specific abbreviations (some words that are considered abbreviations in general text may still be non-abbreviated proper nouns in another language or in a company's own "jargon").

6. CONCLUSIONS

In this article we have presented the implementation of a spoken language interface to e-mail and have justified the choice of a system-directed interaction style by comparing it with the use of a mixed-initiative interface for the same application.

In our tests, we compared the system-driven voice interface with a DTMF interface for the same application. Our results show that, even though the DTMF-driven system receives slightly better scores, the users adapt easily to the speech application and are willing to use it also in the future, since it is easy to use, intuitive and natural.

The outcome of our tests also shows that users that had prior experience neither with DTMF nor with voice interfaces tend to prefer the speech-enabled application.

7. REFERENCES