



# A Paradigm for Mobile Speech-Centric Services

Lars Bo Larsen, Kasper L. Jensen, Søren Larsen, Morten Rasmussen  
 Multimedia Information and Signal Processing (MISP)  
 Dept. of Electronic Systems, Aalborg University, Denmark  
 [lbl,klj,slar,mr]@es.aau.dk

## Abstract

The work presented in this paper describes a new paradigm for speech interaction on mobile devices. A general framework for a distributed architecture is introduced and described. This is followed by a discussion of how to design multi modal interfaces affording spoken input. The solution has been to create an architecture capable of supporting several alternative GUIs, e.g. with spoken input, stylus input or a combination. Speech GUIs are designed entirely without GUI widgets requiring stylus or button input, instead relying on highlighting parts of text to create emphasis and steer the users' attention. This is exemplified through the presentation of a prototype for a Car Rental application.

**Index Terms:** Distributed ASR, client server architecture, multi modal interaction

## 1. Introduction

This paper describes a distributed architecture platform for building mobile speech-centric services. The platform is built as a client-server architecture, with one speech server and one or more application servers. The mobile client can be a Smartphone or a PDA.

The platform rests on three ideas: Distributed Speech Recognition (DSR) is a well-known technique [1] allowing state-of-the-art server based speech recognition engines to become available on relatively low-performance mobile clients. A uniform layout has been designed, with a base window split into a main application frame and a frame with status indicators for connectivity, activity and speech recognition feedback. Finally, an underlying generic application state engine allows a seamless switch between different interaction modes, thus giving the user complete freedom to choose his/her preferred interaction style.

### 1.1. Motivation and Rationale

The motivation behind this work is observations through the recent years of experimental application systems and emerging standards (such as X+V Mobile [2]) for multi modal systems on mobile devices. When studying these systems more closely, it is clear that they have been designed from two different traditions or directions: Either as an evolution from Spoken Dialogue Systems (SDS), with an elaborate dialogue model, reference handling and other mechanisms known from the field of SDS. Alternatively, some systems can be perceived as "speech-enabled GUIs", i.e. traditional Graphical User Interface (GUI) widgets such as form fields, which also support speech input.

One such example is the "MUST" project, where a PDA-based multi modal city tourist guide (for Paris) have been developed and tested. See [3]. The graphical user interface shows a part of a tourist map, with locations of special

interest indicated by icons. The user is expected to tap the icons and ask questions about the site or for navigation instructions. There are no direct visual indications in the GUI that tells the user that speech input is an option or provides guidance of what to say.

Another example is the MATIS project [4]. Here the graphical user interface has been augmented with buttons containing an icon depicting a microphone. While this clearly tells the user that speech is possible, it has the drawback of still demanding a stylus tap-to initiate speech interaction.

These systems and many others like them are pioneers in the area of mobile speech interaction and explorative in their nature, but they illustrate the point made above well - that multi modality is achieved by combining speech and graphics from a well-established starting point of either modality. We see a fundamental problem in this, because the intrinsic capabilities for a specific interaction paradigm (e.g. giving commands) found in either modality might or might not be suitable in a combination with other modalities.

### 1.2. Organisation of the Paper

In this work, we have chosen to look at multi modal interaction from the viewpoint of traditional Human Computer Interaction (HCI) approaches and concepts. This is further discussed in Section 2. In Section 3 the platform architecture is described and Section 4 presents a number of applications we have built upon the platform. Section 5 concludes the paper with a discussion of the implications and the potential of the platform.

## 2. Interaction Styles

In HCI theory, interaction activities are often grouped into four main categories [5]:

- Giving instructions: Issuing commands using e.g. voice commands, a keyboard or selecting options in menus using a stylus.
- Conversing: Interacting with the system as if having a conversation.
- Manipulating and navigating: Acting on objects and interacting with virtual objects
- Exploring and browsing: Finding out and learning things

All four activities can obviously be supported by either speech or graphics. However, a problem might occur, if a multi modal user interface at the same time supports an "Instruction" activity using a pointer (e.g. selecting a departure time with a stylus), while alternatively, the speech modality supports a "Conversing" activity (e.g. speaking a

full sentence specifying day, time and destination). This mismatch contains a potential conflict and might confuse the user and lead him to choose one modality and disregard another throughout the interaction. Indeed, it is our belief that the interaction model promoted by X+V directly encourages a design, where such conflicts will occur in many situations. Ultimately, this might have the effect that the full potential of the speech modality is not realised by the user. The example given above illustrates this. If all visual cues in the GUI are promoting an “Instruction” activity, the user may never become aware of the potentially more powerful “Conversation” activity.

### 2.1. Affordance of Speech

This leads to the issue of affordance. Affordance (in Don Normans’ sense [6]) means “perceived ability to provide” and is in HCI often used to denote what functionalities a given user can perceive from the appearance of an artefact (user interface). Here, speech interfaces have a built-in problem, as unlike GUIs, they do not readily afford a given capability to the user. In other words, the user cannot directly perceive the capabilities of a speech interface, where for example a button in a GUI directly affords pushing. One way to create a high degree of affordance in speech based interfaces is obviously to make use of the visual modality. This can be done by a software tutor [7] or by instructions and examples in help screens, etc., but an obvious solution is to directly incorporate what could be termed “speech affordance” into the interface.

### 2.2. Design principles

When studying the inherent nature of the modalities it becomes clear that speech is non-persistent and linear. This means that, when speech is used for output, the user must pay close attention in order not to miss important information, which obviously puts strains on the users’ cognitive efforts and short-term memory. In opposition to this, the information in a graphical representation will usually stay for the user to inspect until he/she decides to advance the communication at his/her own pace. For input, speech may have the advantage of letting the user address and manipulate objects that are not visible on the screen, and thus allow the user more initiative to e.g. bypass a number of intermediate steps, otherwise required in a direct manipulation interaction.

Based on these observations, we have defined a set of basic design guide lines for the speech-centric user interface. These are briefly explained below, and examples can be seen in more details in the design in section 4.

The most important requirement is that, as the user cannot directly manipulate objects shown on the screen except by speaking, the graphical representations of these objects must not in any way resemble traditional well-known GUI widgets, such as buttons, sliders, drop-down menus, hyperlinks, text fields, etc. Or, put differently, the GUI must not afford direct manipulation. Furthermore, in this study we will not use spoken output.

Instead, the graphical representations of objects must encourage users to address them verbally. This is facilitated through a larger degree of verbosity than normally used in GUIs and especially by the use of emphasis. Emphasis can be achieved in several ways, such as **bold face**, *italics*, underlining and **different colouring** of the text segments (words) that the system will understand and react upon. By

using emphasis the user will be steered towards using identical or similar terms and constructs in an intuitive manner, without explicitly being told to do so.

## 3. Platform Architecture

As mentioned above, the architecture of the proposed platform supports a distributed architecture with two or more servers. This is shown in Figure 1 below.

### 3.1. The Graphical Interface Modules

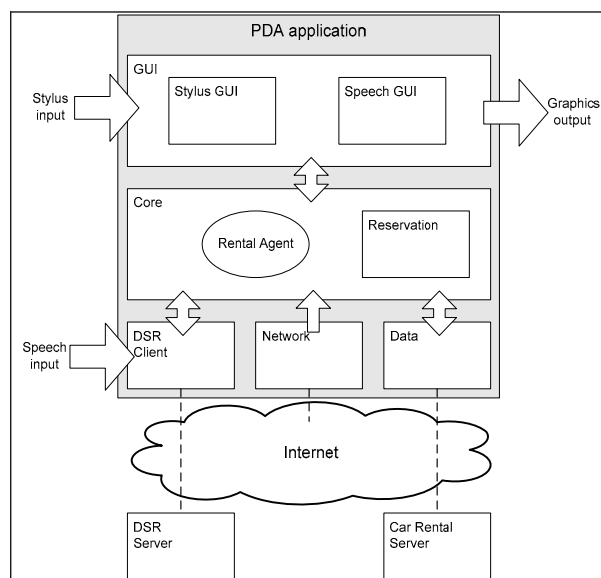


Figure 1. Distributed Platform Architecture exemplified with the “Car Rental” application, described in Section 4

The main principle is the separation of the GUIs from the underlying Core engine in the client. By introducing this separation, it becomes possible to implement several GUIs independently. In the present case (the Car Rental application described below) the two distinct GUIs are realised, one for speech and one for stylus interaction. In other cases, e.g. a mixed-mode or a button-press GUI might be designed. However, as all information about of the states of the interaction is kept in the Core layer, a seamless switch between each GUI can be performed at any time. Note that “GUI” refers to any number of individual screen layouts. Thus, the user has complete freedom of choice, according on his personal preferences or e.g. a noisy or public environment.

### 3.2. Distributed Speech Recognition

As illustrated in Fig. 3, the DSR system [8] is developed on the basis of the ETSI-DSR advanced front-end (AFE) [1], [9] and the SPHINX IV recognizer [11]. The advanced front-end client-side module extracts noise-robust Mel-Frequency Cepstral Coefficient (MFCC) features which together with Voice Activity Detection (VAD) information are encoded sequentially and packed into speech feature packages for network transmission.

At the server side the received speech packages are processed by the advanced front-end server-side module. First, on the detection of transmission errors, error concealment is conducted for feature reconstruction. Then, the error-corrected speech feature packages are decoded into a set of

cepstral features and VAD information. Subsequently, the features are processed by the SPHINX IV speech recognizer. The recognizer presents its result (either the best or N-best results) at the utterance end, detected by the VAD information, and transmits it back to the client.

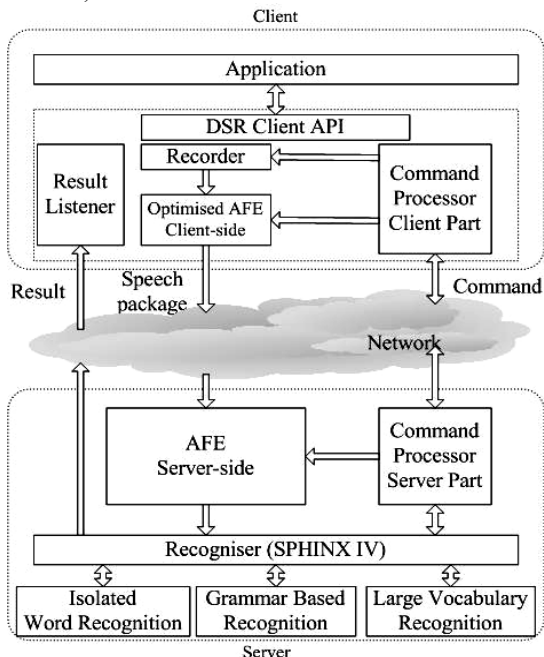


Figure 2. DSR implementation

Figure 2 shows in details the architecture of the DSR module. This is in turn integrated into the overall platform shown in Figure 1 above.

Both the DSR and the application servers can be placed anywhere in the network in a manner similar to e.g. X+V applications. In the present case, the DSR channel requires a bandwidth of 5.6 Kbit. The bandwidth of the application channel obviously differs from one application to another, but unless large quantities of graphical objects are downloaded, this is most often quite limited. Thus, the network requirements can in most cases be fulfilled by a standard GPRS connection.

#### 4. Examples of Applications

A number of applications have been designed and implemented using the platform. Initially, an information retrieval application for Danish soccer news was developed to demonstrate how information retrieval tasks can be supported by spoken queries [12]. This was later adapted to an English version with information from the Premier League.

A more elaborate “Car Rental” application [13], [14] has been developed recently to illustrate the design principles described in Section 2.2 above. This section presents the Car through a number of screen shots of the “stylus GUI” vs. the “Speech GUI”.

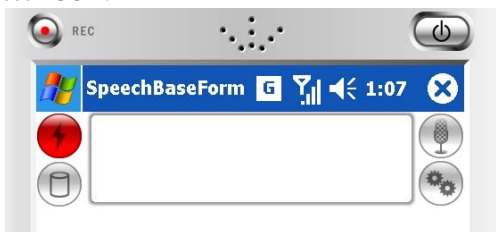


Figure 3. Upper part of the screen with status indicators

Figure 3 shows the topmost part of the generic Speech GUI. The main field contains the speech recognition feedback and the icons to the right and left indicates the states of the recogniser, network connections to the servers, etc. Generally, the intention is that the upper part of the screen will always be showing this information. However, this takes up approximately 20% of the screen, which might be undesirable in some cases. We have experimented with the upper part sliding upwards off the screen after a delay of 5 seconds. This seems to work satisfactorily and it will automatically reappear when needed.

As mentioned in Section 2.2 we use highlighting to create emphasis to draw the users’ attention towards particular salient information. In Figure 4 below, key concepts are highlighted in the speech recognition feedback pane on the top (“vælge forsikring” / “choose insurance”). Although the whole sentence is recognised, only the semantically salient information is highlighted and has led to a pop-up window showing various options for buying additional insurance. Highlighting is also used here to indicate the concepts that the user can select or deselect. Of course the emphasis also leads the user to choose a particular phrasing, thus supporting the recognition task. At the bottom of the screen another pane is shown, where the status of the dialogue is summarised. Highlighting is used to indicate that the user still has not chosen an insurance type.

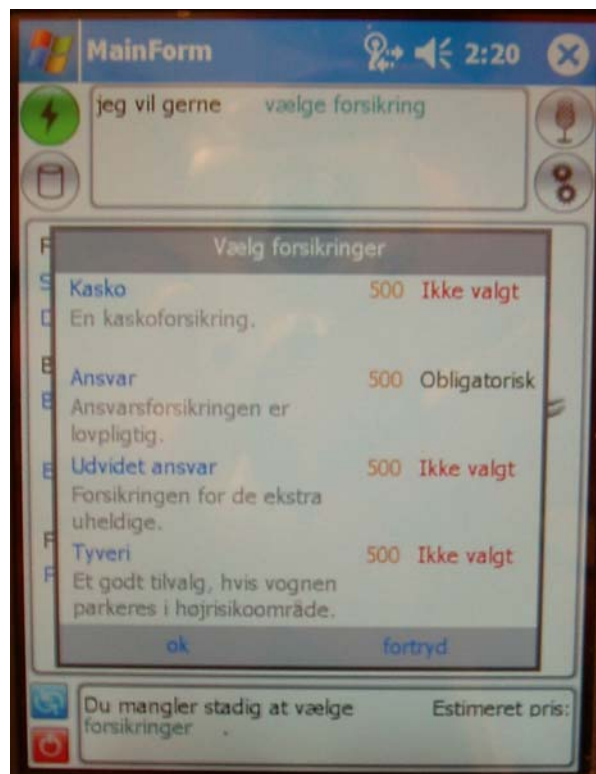


Figure 4. Screenshot showing the use of Emphasis through application of coloured text

It should be noted that the colours are chosen rather bright to better show up in the screen shots. Also note that the screen contains no widgets looking like buttons, links, menus, etc., thus affording only spoken input. However, the user can switch to a traditional stylus-based interaction GUI simply by pressing the button at the lower left corner. An example of this is shown in Figure 5 below. The differences are obvious,

as this screenshot shows the selection of a car, with “next”, “previous” buttons, a dropdown menu, etc. Note the mode shift button is still visible in the lower left corner.



Figure 5. Screenshot showing a traditional GUI screen in a wizard-like dialogue

## 5. Discussion and Conclusion

We have presented an alternative to the way multi modal interfaces have been perceived and designed up till now. Our focus on the affordance of the speech modality and the interaction activity types has led us to a higher degree of separation between speech and non-speech input than commonly seen. Instead of allowing interface widgets to support both non-speech (e.g. pointing) input and speech input in parallel, we have completely removed any elements affording non-speech input. Instead we create emphasis by the use of highlighting to focus and steer the user towards the goal.

We have created an architecture to support this notion, based on a core GUI engine, which holds the states of the separate GUIs and thus facilitates seamless transition between the GUIs at any point in the interaction.

We are well aware that there is an overhead associated with our approach. For example, more screens must be designed and implemented. Likewise, the highlighting of key concepts, both in the recognition results and the GUIs is not without difficulties. On the other hand, each screen may be more simple and easy to design and if a total separation of speech and stylus interaction is sought, this will greatly reduce the need for synchronisation between modalities, which is a notoriously hard task.

An elaborate end user test suite to determine the optimal use of emphasis and user preferences is planned for the near future, but has not been carried out yet.

## 6. Acknowledgements

This work was supported by the Danish Government through a grant for the CNTK (Center for Network and Service Convergence) project. The authors wish to thank Zheng-Hua Tan and Haitian Xu for their work on the DSR speech recogniser.

## 7. References

- [1] ETSI Standard ES 202 212. “Distributed speech recognition; extended advanced front-end feature extraction algorithm; compression algorithm, back-end speech reconstruction algorithm”, November 2003.
- [2] VoiceXML Forum Specification: “Mobile X+V 1.2”, published 2005:  
<http://www.voicexml.org/specs/multimodal/x+v/mobile/12/>
- [3] L. Almeida1, I. Amdal, N. Beires, M. Boualem, L. Boves, E. den Os, P. Filoche, R. Gomes1, J. E. Knudsen, K. Kvale, J. Rugelbak, C. Tallec, N. Warakagoda: “The MUST guide to Paris: Implementation and expert evaluation of a multimodal tourist guide to Paris” In proceedings of ITRW Workshop on Multi-Modal Dialogue in Mobile Environments Germany 2002
- [4] J. Sturm, I. Bakx, B. Cranen, J Terken: “Comparing the Usability of a User Driven and a Mixed Initiative Multimodal Dialogue System for Train Timetable Information” In proc. of Eurospeech 03, Geneva, Switzerland 2003
- [5] Helen Sharp, Yvonne Rogers, Jenny Preece "Interaction Design: Beyond Human Computer Interaction, 2nd Edition" January 2007, ISBN: 978-0-470-01866-8
- [6] Donald Norman, The Design of Everyday Things, 1988, ISBN 0-465-06710-7
- [7] Jaakko Hakulinen, Markku Turunen, & Esa-Pekka Salonen. “Software Tutors for Dialogue Systems”. In Proceedings of Text, Speech and Dialogue (TSD 2005), LNAI 3658, Springer, pages 412-419.
- [8] Tan, Z.-H., Dalsgaard, P. and Lindberg, B.: Automatic speech recognition over error-prone wireless networks, Speech Communication, 47(1–2), 220–242, 2005.
- [9] 3GPP TS 26.243: ANSI-C code for the Fixed-Point Distributed Speech Recognition Extended. Advanced Front-end, December, 2004.
- [10] Xu, H., Tan, Z.-H., Dalsgaard, P., Mattethat, R. and Lindberg, B.: A configurable distributed speech recognition system, Biennial on DSP for in-Vehicle and Mobile Systems, Sesimbra, Portugal, Sep. 2005.
- [11] The CMU Sphinx Group Open Source Speech Recognition Engines.
- [12] Brøndsted, T., Larsen, H.L., Larsen, L.B., Lindberg, B., Ortiz-Arroyo, D., Tan, Z.-H., Xu, H., “Mobile Information Access with Spoken Query Answering” in COST278 Final Workshop on “Applied Spoken Language Interaction in Distributed Environments” ASIDE, Aalborg, Denmark, Nov 2005
- [13] J.J. Jensen, K. L. Jensen, L.B. Larsen, S Larsen: “The Car Rental Service: Design and Prototyping” Technical Report (CNTK), Aalborg University, April 2006 ISBN: 87-90834-94-1. ISSN: 0908-1224
- [14] K. L. Jensen, L. B. Larsen. S. Larsen, M. H. Rasmussen “Implementation of the CNTK Car Rental Service” Technical Report (CNTK), Aalborg University, November 2006 ISBN: 87-90834-97-6. ISSN: 0908-1224