From Vocal to Multimodal Dialogue Management

Miroslav Melichar
École Polytechnique Fédérale de Lausanne (EPFL)
Artificial Intelligence Laboratory (LIA)
CH-1015 Lausanne, Switzerland
miroslav.melichar@epfl.ch

Pavel Cenek
Masaryk University
Faculty of Informatics
Laboratory of Searching and Dialogue
602 00 Brno, Czech Republic
xcenek@fi.muni.cz

ABSTRACT

Multimodal, speech-enabled systems pose different research problems when compared to unimodal, voice-only dialogue systems. One of the important issues is the question of how a multimodal interface should look like in order to make the multimodal interaction natural and smooth, while keeping it manageable from the system perspective. Another central issue concerns algorithms for multimodal dialogue management. This paper presents a solution that relies on adapting an existing unimodal, vocal dialogue management framework to make it able to cope with multimodality. An experimental multimodal system, Archivus, is described together with discussion of the required changes to the unimodal dialogue management algorithms. Results of pilot Wizard of Oz experiments with Archivus focusing on system efficiency and user behaviour are presented[1].

Categories and Subject Descriptors

H.5.2 [**Information interfaces and presentation**]: user interfaces—*graphical user interfaces, interaction styles, natural language, prototyping, voice I/O*

General Terms

Algorithms, Design, Experimentation, Human Factors

Keywords

Multimodal systems, dialogue systems, dialogue management, rapid dialogue prototyping, Wizard of Oz, human computer interaction (HCI), graphical user interface (GUI)

[1]The research presented here is part of the Swiss NCCR on "Interactive Multimodal Information Management" (IM2, `http://www.im2.ch`), funded by the Swiss National Science Foundation.

ICMI'06, November 2–4, 2006, Banff, Alberta, Canada.

1. INTRODUCTION

In the last few years, research on unimodal, voice-only dialogue systems has slowly moved towards multimodal, speech-enabled systems [12, 1, 3]. It is believed that use of multimodality will increase the robustness and flexibility of such systems and allow for more natural communication between the system and users [9].

This seems to be intuitively true since offering the users several ways of providing input allows them to use the "most convenient" modality in a given situation. For example, the most convenient way of selecting a date could be clicking on the image of a calendar, while the easiest way of finding more complicated information such as *Who attended all of the meetings* could be to type it as a query in natural language. Moreover, the choice of modality can also be a matter of personal preference and the user's experience with the system (e.g. command-line user vs. GUI user).

However, before multimodal systems can offer their full potential to users, several important research problems must be solved. One of them is the question of how a multimodal interface should look like in order to make the multimodal interaction natural and smooth, yet remain manageable from the system perspective. Another concerns the design of efficient algorithms for multimodal dialogue management.

Our approach to addressing these problems is presented in this paper. It relies on adapting an existing unimodal, vocal dialogue management framework to cope with multimodal input and output. As a testbed, we created Archivus – a multimodal dialogue system for accessing a multimedia database of recorded and annotated meetings [5].

We have carried out a series of pilot Wizard of Oz experiments to see how users would react to and interact with the multimodal Archivus interface. One of the goals of the experiments was to collect and evaluate language-related data, and another was to test the efficiency of the underlying dialogue strategy in a multimodal application (as the original dialogue management framework was designed for a unimodal system).

The rest of this paper is organized as follows: Section 2 describes the unimodal dialogue management framework on which we based our work. Section 3 introduces our multimodal system Archivus and Section 4 describes the changes to the unimodal dialogue management framework that were required in order to support multimodality. Section 5 presents our experimental environment and various statistical results regarding the multimodal behaviour of the users. Some concluding remarks close the paper.

2. UNIMODAL DIALOGUE MANAGEMENT

The unimodal dialogue management approach that has been taken as a basis for the multimodal dialogue management of the Archivus system has been developed as a part of the rapid dialogue prototyping methodology (RDPM) [2]. RDPM has been tested in several projects, such as InfoVox[2] – an interactive vocal system for providing information about restaurants, and INSPIRE[3] [7] – a dialogue system for the vocal control of domestic devices within a Smart-Home environment.

The RDPM was developed for frame-based dialogue systems [6] that help users find one or more objects that best correspond to their needs (e.g. the right restaurant) from a potentially large set of objects. The needs (hereafter called *search criteria*) are progressively expressed by the user during interaction with the system and are internally represented as a set of `name:value` pairs (hereafter called *semantic pairs*). An example of a set of semantic pairs is shown in Figure 1. The values of the semantic pairs are inserted into slots with the corresponding name. There is a hierarchical, tree-like structure built above the slots that reflects the structure of the dialogue domain (hereafter called the *task model*). A simple task model is shown in Figure 2.

first_name:	Peter
family_name:	Smith
year:	2006
month:	8
day:	17

Figure 1: Semantic pairs partially describing a reservation.

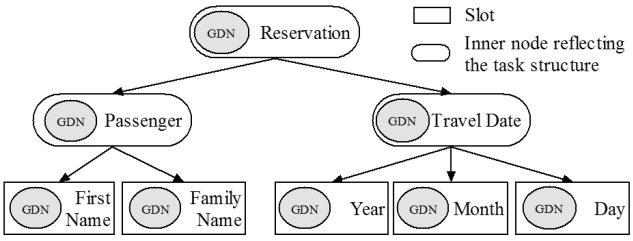


Figure 2: A simplified task model in the flight reservation domain, with associated GDNs.

2.1 Generic Dialogue Node (GDN)

In the RDPM, there is a *generic dialogue node* (GDN) associated with each node in the task model. The role of the GDN is to perform the interactions with the user that are required to obtain a valid value for the node. For the GDNs associated with leaf nodes, the value obtained is directly the value of the slot. For the higher-level nodes the value is any combination of values from the leaves (slots) covered by that node. The situation is depicted in Figure 2.

There are different types of GDNs available that encapsulate different types of interaction according to the nature of the values. For example, one type of GDN allows the user to directly specify a value for the associated slot (e.g. selecting a day of week), while another allows the user to browse through a potentially long list of values and select one of them.

2.2 Dialogue Strategies

The term *dialogue strategy* refers to the decision of the dialogue manager about the next step of the dialogue. In the unimodal dialogue management framework, a two-tier dialogue strategy model is used [2], distinguishing *local dialogue strategies* from *global dialogue strategies*.

The purpose of local dialogue strategies is to handle problems related to the interaction *within* a particular GDN, such as handling requests for help, speech recognition failures (no match), situations when no input is provided, etc. The local strategies are carried out by the local dialogue manager which is a part of each GDN.

Global dialogue strategies, which are implemented by the global dialogue manager, are responsible for navigation *between* GDNs (i.e. dialogue focus selection), for handling situations when no object in the database corresponds to a user's requirements (overconstrained situation), or when two distinct values are inserted into one single slot (incoherency).

Our dialogue prototyping methodology is referred to as *rapid* because the dialogue designer only needs to specify the task model, to associate a suitable GDN with each of the nodes defined in the task model and to configure parameters such as the prompts and speech recognition grammars for each GDN. The other aspects of the dialogue system are handled automatically. In particular, the dialogue designer does not need to define the dialogue strategies controlling the interaction with the user. Additionally, a novel aspect of our dialogue prototyping methodology is that it allows for iterative refinement of system parameters in the early stages of development by means of Wizard of Oz experiments, which are in fact an integral part of the dialogue management framework.

2.3 Wizard of Oz Experiments

The RDPM provides built-in support for creating and using Wizard of Oz (WOz) experiments [4] and it is possible, at any stage of the development, to run the dialogue manager autonomously, or connected to a WOz assistance platform. In the latter case, functionalities that have not yet been fully implemented can be simulated by a hidden human operator called the wizard. To do this, the wizard uses a Wizard's Control Interface (WCI) which is accessible from a computer called the wizard's control station.

The main role of such experiments is to allow for the acquisition of experimental data about the behaviour of users when interacting with the system, without having to fully implement every aspect of the dialogue management. Such information is particularly valuable in the early stages of system design.

When the system is running in the WOz mode, the dialogue system data flows are redirected to the wizard's control station and the wizard can check or modify them, or even

[2]InfoVox: Interactive Voice Servers for Advanced Computer Telephony Applications, funded by Swiss national CTI 4247.1 grant

[3]Inspire: INfotainment management with SPeech Interaction via REmote microphones and telephone interfaces, IST-2001-32746

create new pieces of data if necessary. Once the wizard is satisfied with the data, (s)he sends it back to the dialogue system and the dialogue processing can continue.

The WCI allows for simulation or supervision of the input (i.e. the semantic pairs produced by the natural language understanding module or text transcription of user utterances) and supervision of the decisions made by dialogue strategies. The possibility of (re)starting or stopping the entire system is also available.

3. A MULTIMODAL SYSTEM: ARCHIVUS

As a testbed for validating our approach and various scientific hypotheses related to it, we created Archivus.

Archivus is a multimodal dialogue system for accessing a multimedia database of recorded and annotated meetings [5]. Specifically, the database contains the original video and audio from the meetings (recorded in special meeting SmartRooms), electronic copies of all documents used or referred to during the meetings, as well as electronic copies of handwritten notes made by participants, and a text transcript of the meeting itself. In order to facilitate retrieval of potentially complex information, annotations have also been made on the data, specifying elements such as dialogue acts, argumentative structure and references to documents, as well as metadata such as the date and location of the meetings and information about the meeting participants.

In Archivus, the task of the users is to retrieve specific pieces of information about what happened in different meetings, for example topics that were discussed, decisions that were made, documents that were presented, or people who were active in proposing ideas. Archivus uses a library metaphor (bookcases, books, etc.) to facilitate the understanding of how to access this information – see Figure 4(a). Meeting transcripts can be browsed page by page or viewed as a video. But, if the user is interested in a specific segment of a meeting or a document that was presented, Archivus provides users with the possibility to specify search criteria, either through manual selection in the graphical interface or through natural language requests, such as *"Show me where they discussed about meeting room furniture"*. The system uses this information to point out to the user the relevant meetings and meeting segments that correspond to the specified criteria.

The Archivus interface is meant to be flexibly multimodal, meaning that users can interact with the system either unimodally, using any of mouse, keyboard, speech or touchscreen, or multimodally, using any combination of the available modalities. Graphical output to the screen is also combined with spoken output.

4. EXTENSIONS TO MULTIMODAL DIALOGUE MANAGEMENT

As already mentioned, the initially unimodal dialogue management framework outlined in Section 2 has been extended to support multimodality in the Archivus system, where spoken and typed natural language, mouse and touchscreen are available as input modalities and speech, graphics and video on the screen as output modalities. The extended framework seems to be general and may be applied to other systems (equipped with screen) as well.

Although several adaptations of the original dialogue management framework were needed, its main pillars have been preserved – (1) the hierarchical task model with a GDN associated to each node, (2) the two-tier dialogue strategy model,(3) the representation of user's input as a set of semantic pairs and (4) the built-in support of WOz experiments.

The following subsections describe in detail the required extensions and modifications, including the motivations justifying these changes.

4.1 Multimodal Generic Dialogue Node (mGDN)

In the original vocal dialogue management framework, GDNs are responsible for prompting the user, processing their input and representing it as a set of semantic pairs. While this general principle has been kept, the GDNs have been extended to *multimodal GDNs* (mGDNs). In addition to its original content, the mGDN also defines grammars for written natural language input and the voice prompts used as system output have been extended to multimodal prompts that define both audio and visual representation of the prompt (for example a text that is to be spoken out and text that is to be simultaneously displayed on the screen).

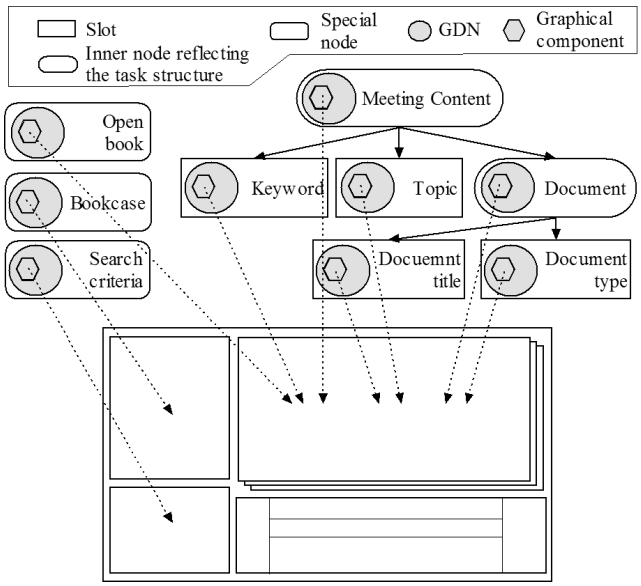
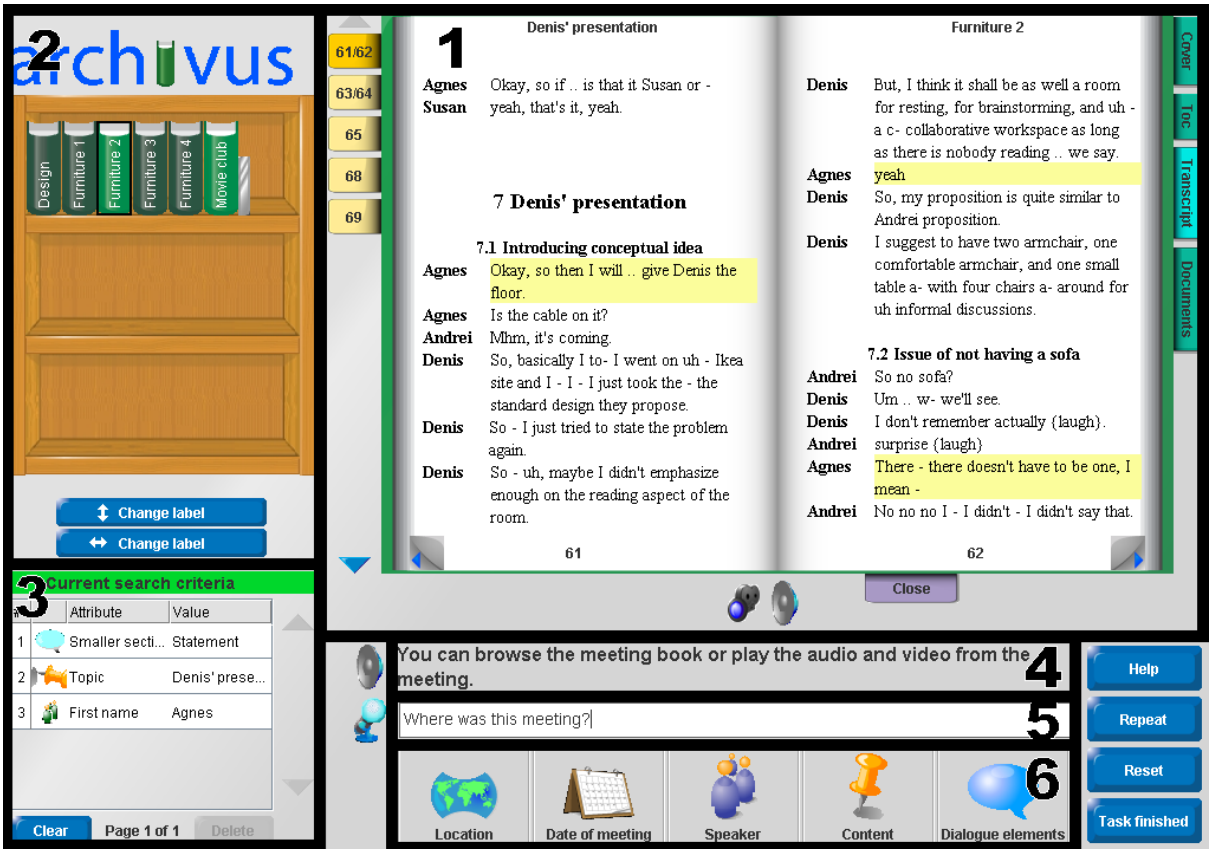


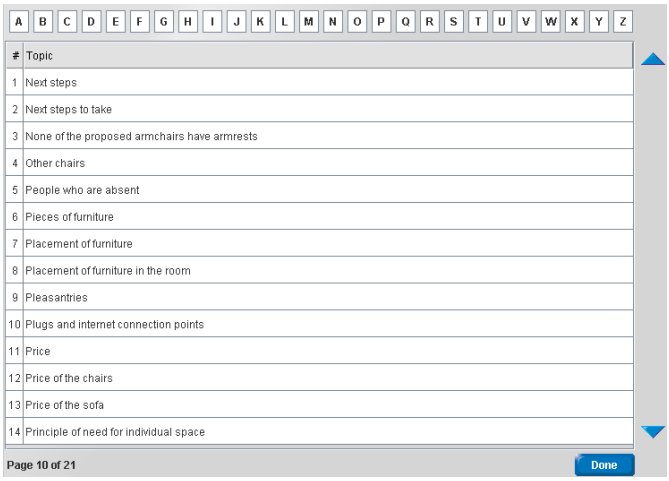
Figure 3: A part of the task model representing the Archivus system, with multimodal GDNs and the associated graphical components.

Since our multimodal system uses a screen for interaction (while the original multimodal system did not), there is a graphical component associated with every mGDN, as depicted in Figure 3. The graphical component is responsible for displaying the graphical user interface specific to that mGDN, thus allowing mouse/screen communication with the mGDN (in addition to the original spoken interaction). Several graphical components have been created, each having a different layout and being able to display different types of data.

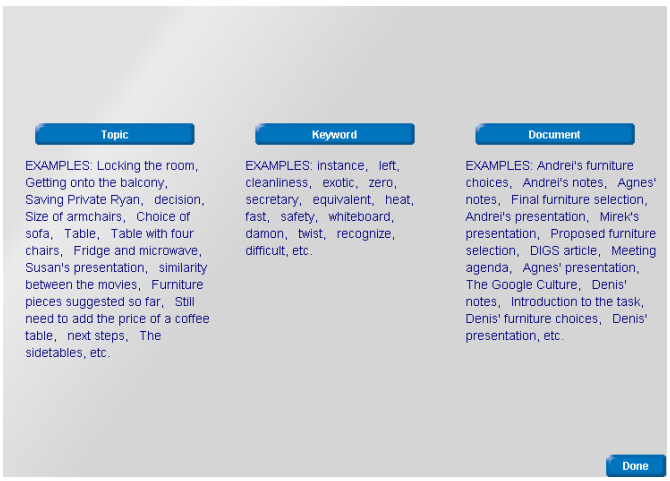
An example of a graphical component allowing the user to select a value for an mGDN from a list is the *list layout* component shown in Figure 4(b). The users can scroll through the list using the arrows on the right side and select



(a) Archivus user interface: 1. graphical component representing the content of the meeting in the form of an open book with relevant results highlighted; 2. graphical component displaying all meetings as books on a bookshelf, relevant meetings are displayed as light green books; 3. graphical component showing search criteria entered by the user; 4. system prompt; 5. input bar for typed natural language; 6. (main) criteria selection buttons – selecting any of these buttons displays currently accessible values for the selected search criterion in the central part of the screen (1).



(b) Graphical component with the *list layout* allowing the user to select the *value* for a search criterion from the list.



(c) Graphical component with a *cue layout* allowing the user to select the *type* of search criterion (s)he wants to specify by clicking on the corresponding button. Some examples of values for each type of specific search criterion are also displayed and the user can enter a value directly by voice or by typing it using the system's typed natural language input bar.

Figure 4: Archivus system screenshots

an item by clicking on it. They can also use the buttons at the top to jump to the first item starting with the specified letter. The *Done* button finishes the interaction with the mGDN without selecting a value and the system then changes focus to another mGDN. An mGDN that uses this layout (typically a leaf in the task model) also includes a grammar for controlling the graphical component by voice. For example, the user can scroll through the list using the vocal commands "up" and "down" and select a value by saying the corresponding number which is specified in the first column. Any values, including the ones that are not currently displayed, can also be directly spoken or typed.

Another example of a graphical component is the *cue layout* shown in Figure 4(c). This graphical layout is typically used by mGDNs associated with inner nodes having leaves as children. The user can explicitly change focus to any child mGDN by clicking on the corresponding button. Some examples of values for each of the child nodes are also displayed and the user can enter the value directly by voice or by typing it in using the system's typed natural language input bar.

Some other examples of graphical components are the component displaying a map with scrolling and zooming capabilities for location selection, or a calendar for date selection. There may also be application-specific graphical components, such as the bookshelf and open book in Figure 4(a) in the case of the Archivus system.

4.2 Processing User's Input

The graphical components of several mGDNs may be displayed on the screen at the same time, as can be seen in Figure 4(a). At any given time only one of them is highlighted to indicate that it is linked to the mGDN in focus – the mGDN that is currently responsible for communication with the user and for processing their input through any available modality.

The notion of a single mGDN in focus is important for resolving ambiguous natural language input. The mGDN in focus is the first one that gets natural language input for processing. Only after it is not able to process the utterance or resolve some references, other mGDNs (called *active*) get their chance. The order in which mGDNs process the input is derived from their distance in the mGDN hierarchy from the mGDN in focus and preference is given to the mGDNs associated with graphical components currently displayed on the screen.

In the multimodal case, this technique is even more important since commands for controlling the GUI are often applicable to several elements on the screen. For example, if there are several mGDNs on the screen that display scrollable list of items, user input *"Go down"* would be ambiguous, as it would not be clear which list should be scrolled. In our case, the mGDN in focus would perform the command. Other mGDNs can only process the command if it was not applicable to the mGDN in focus.

As already mentioned, the function of a graphical component is not only to display the GUI of the associated mGDN, but also to react to mouse actions. In particular, as a reaction to a mouse click, the graphical component produces one or more semantic pairs. The produced semantic pairs represent the operation performed on the graphical component in the same way as they represent the meaning of a spoken or typed natural language query or command.

Note that at this point the graphical component is not yet updated in any way. The generated semantic pair(s) are processed by the mGDN or global dialogue manager and only after this is done the mGDN updates the state of associated graphical component. This mechanism makes it possible for the wizard to intercept and validate the operation, fuse input from multiple modalities and interpret its meaning in combination.

4.3 Multimodal Input Fusion

When extending our unimodal system to become multimodal, the problem of multimodal input fusion had to be considered. Multimodal fusion is one of the most active research areas in the field of multimodal systems. It is a very difficult task since it includes problems such as reference resolution and temporal aspects. So far, several approaches and algorithms have been proposed [8, 10] but the problem remains an open issue.

To make multimodal fusion manageable in our framework, we decided that each mGDN will process the input separately, without taking the existence of other mGDNs into account. Our assumption is that the fusion algorithm for each individual mGDN will be feasible to design. Note that this approach automatically eliminates the possibility of references to entities from other mGDNs. However, it does not limit mixed initiative since the utterance can be processed by several mGDNs, one after another, each processing a part of the utterance.

As a starting point, we kept the notion of dialogue turns as in the voice-only case and used the following simple fusion algorithm:

Semantic pairs produced by any of the available modalities in a single dialogue turn are merged into a single set and then processed by the following steps, very similar to the ones used in the unimodal case:

1. The semantic pairs are first sent to the WCI for supervision (if the WOz control mode is used).

2. The semantic pairs (possibly modified by the wizard) are processed by the mGDN *in focus*. If the semantic pairs represent a task that can be handled by the local dialogue strategy of the mGDN, the local dialogue strategy is carried out. In addition to the local dialogue strategies mentioned in Section 2.2, the multimodal system introduces a new class of local dialogue strategies which concern updates of graphical components performed by the associated mGDN (e.g. scrolling one page down in the list or jumping to a specific section of the open book). In general, this class includes GUI-related operations that do not lead to a slot value selection nor to a focus change.

3. If the semantic pairs cannot be processed by the local dialogue strategies of the mGDN in focus, they are passed to the global interaction manager. The global interaction manager can itself process the semantic pairs if they are directly meaningful to the global dialogue management strategy (e.g. when an explicit focus change is requested) or it offers them to other mGDNs that are not in focus, but are active (mixed initiative support).

4. The active mGDNs receive the yet unprocessed semantic pairs. The pairs are processed by the mGDN if they

represent a local dialogue strategy, e.g. scrolling, request for help, etc.

Before designing a more complex multimodal input fusion algorithm, we decided to arrange a series of Wizard of Oz experiments that would show what kind of multimodal interaction is typical for Archivus. During the experiments, the wizard could easily simulate more complex multimodal fusion, including resolving referential expressions. However, the experiments showed that no complex multimodal behaviour occurred. Although users used all available modalities, they did so almost always sequentially. The reason could be that the information-search task, as opposed to various map-navigation tasks, does not induce "true" multimodal behaviour, or that our system, which uses common office hardware, does not induce such behaviour. Consequently, there is currently no need for more complex multimodal fusion in the case of Archivus.

4.4 User-Driven Dialogue Strategies

In systems that only permit vocal input, the user typically expresses some initial wishes at the very beginning of the interaction and the system then progressively asks for missing information, guiding the user towards the goal of the interaction (i.e. to find objects in the database satisfying the user's needs). It is important to guide the users in vocal-only systems, as they may not know what piece of information they need to supply to the system to reasonably limit the number of objects satisfying their needs. Each time the user provides some new information, the vocal dialogue manager analyses the current solution set and selects the next most appropriate piece of information the user should be asked for. This is done by changing the focus of the interaction to the GDN associated with a slot that is associated with this piece of information.

In multimodal systems equipped with a screen, users have visual feedback on the current context of the interaction, for example, they can quickly *see* what type of information the system requires, and they may have an instant global view of the current solution space as well as of the constraints that have been provided to the system so far. In such situations, users have a better understanding of the current context of the interaction and it is therefore reasonable to assume that they will prefer to participate more actively in the interaction.

The system GUI gives the user a natural possibility to explicitly change the focus (i.e. to select the next mGDN) by clicking, and therefore the multimodal dialogue manager must be able to handle this change. This was an extension to the original vocal dialogue management strategy, where such a feature seems to be very unnatural as it would correspond to user utterances like *"Now I want to tell you the speaker of the meeting"* or *"Ask me for the speaker in the meeting"*. The explicit focus change feature is not needed in vocal dialogue systems with mixed initiative since speech can express directly value(s) different from those expected in the current dialogue step. For example, if the system is asking *"What was the location of the meeting?"*, the user may vocally answer *"It was in Lausanne and John was speaking"*, while this is not possible (without explicit focus change) by mouse pointing when only a list of locations is displayed on the screen.

Another specificity of the Archivus multimodal system is that the original strategy for selecting the next GDN in focus (after a value for the current one was obtained) was negatively perceived by users. We see two possible reasons for this problem: (1) the graphical interface allows users to clearly see the hierarchical structure of the task, and the user is confused when the system automatically selects an mGDN in another part of this hierarchy (the behaviour of the system does not seem to be consistent and stable), and (2) the task that is solved using the Archivus system is more complex (larger number of slots) than tasks that can be solved by voice only systems, making the automated selection of the next mGDN (based on the current solution set) partly inappropriate from user's perspective, as the system may ask for information that the user does not know or has no preferences for.

It was therefore decided to make the dialogue management strategy more passive: after a value for the current mGDN is acquired (or the user indicates that he does not want to provide a value by selecting *Done*), the focus is automatically changed, but only to the closest ancestor of the current mGDN in the mGDN hierarchy such that the ancestor covers at least one slot whose value needs to be provided by the user.

The hierarchical structure of the task is actually presented to the user as a sort of hierarchical menu. From the perspective of the user, our updated dialogue strategy therefore corresponds to going back to one of upper menu items in the menu hierarchy.

4.5 New Role of System Prompts

When extending the unimodal vocal dialogue prototyping methodology for multimodal systems, the design of system prompts has to be addressed differently. In a voice-only system, prompts are the only means of conveying information to the user. However, in the case of Archivus, users are no longer interacting with the mGDN in focus in isolation. Instead, they can see several other active mGDNs displayed on the screen at the same time and each of these mGDNs can contribute with some useful information. In addition to the GUI for entering value(s) for the slot(s) associated with the mGDN in focus, the user can also easily see for example the submitted search criteria, solutions satisfying these criteria, and the signalization of an overconstrained situation. This gives the user a better idea of the current context of the conversation compared to voice-only systems.

Since the original role of the prompts, i.e. request for user's input, is often taken over by the graphical representation, the vocal prompts can be either omitted entirely in some situations or used to provide the user with context-dependent suggestions and advice. For those reasons, we make the distinction between "information requesting" prompts and "advising" prompts. For the system developer, it is not as easy to identify the context-dependent advising prompts and integrate them into the multimodal dialogue model as it was with the requesting prompts in unimodal models. Typically, an advising prompt should be used if the user ignores some important signal given by the graphical interface. For example, in Archivus, meetings relevant to user's current request are displayed as light green books in the bookshelf, while the irrelevant books are dark green. When the user tries to open a dark green book, an advising prompt should be played informing the user that this meeting does not contain any relevant information.

An advising prompt can be also useful in situations when

the default prompt chosen by the system is redundant. For example, if the user has interacted with the system for a while, (s)he does not need to hear the information provided by the default prompt again. Instead, an advising prompt can point out an interesting aspect of the system, teaching the user to use the system more effectively. If there is no suitable advising prompt available, it is often better to omit playing a prompt altogether.

In order to discover the type of prompts that are relevant for a multimodal dialogue model, evaluating the system with users is crucial. For that purpose, we extended the Wizard of Oz control interface capabilities with a new functionality which allows a wizard to dynamically control the system prompts. If the wizard identifies a situation in which the prompt proposed by the system is not optimal, (s)he can replace it with a more appropriate one, either by selecting it from a predefined list or by typing a new prompt on the fly. Each new prompt that is created during interaction is added to the list of possible prompts, so that when a similar situation occurs, the wizard can reuse the already produced prompt. All changes are logged so that they can be analyzed later and used for improving the design of the dialogue model to better fulfil the requirements of multimodal systems.

When changing the prompts, the wizard must be consistent in his behaviour in order to produce results that can be later generalized into an algorithm or at least some guidelines for system designers. Therefore, the task was performed by one and the same person during the whole experiment period. This person was familiar with the system, knew which questions the users were working on and had enough experience with those questions from previous pilot studies to be able to predict quite well what the user would do in a particular situation.

Since advising prompts should be played only when certain circumstances occur, a notion of conditional prompts should be introduced into the dialogue model description language. Each advising prompt would be associated with a condition describing the situation in which this prompt should be used. If the condition is satisfied, the associated prompt replaces the default one. Identifying what conditions are relevant is an open research issue.

4.6 The Wizard's Control Interfaces

The presence of several input and output interaction channels forces several significant changes to the Wizard's Control Interface. Early experiments with Archivus showed that the cognitive load of a wizard doing all the tasks that require a wizard's intervention (speech recognition, natural language understanding and prompt generation), was too high. As a result, we have decided to use two wizards, one that processes user input and another that controls the output prompts. Each wizard has a control interface designed for their specific task.

For the Input wizard, it is important that the control interface is designed in such a way that the time required for processing input from the different modalities is well balanced. From the user's perspective, the processing time corresponds to the system reaction time, and if the user notices that one modality leads to a longer reaction time than another, this may influence their modality preference, which in turn influences the validity of the data that is produced during the experiments. To overcome this problem, we have made several improvements to the Input Wizard control interface to allow the wizard to react more efficiently to language input, for instance by introducing "short-cut" buttons for frequently used voice commands, and by implementing a filter mechanism for quickly selecting the appropriate semantic pairs (representing the user's input) from a long list.

The Output Wizard has a control interface that was not a part of the WOz environment of the original unimodal dialogue prototyping methodology. This wizard also needs to be very efficient in their task, which means that (s)he should avoid typing new prompts as much as possible and rather select them from a predefined list. To be able to produce this list before the actual experiments start, a small pilot experiment with a few users was necessary. These users gave some indication about the most frequent situations in which a prompt may need to be changed, so that when the large-scale experiments began, very few new prompts needed to be created at run-time.

In terms of hardware, the wizard's control station is significantly more complex than the one used for the original, voice-only configuration. In addition to the computers that are needed for running the WCIs, the wizards also need two computers (connected over a network to the user's room) for receiving important information about the user which the wizards use to perform their simulation tasks. On one screen they observe the user's desktop, which helps them to follow the user's interactions and predict the next action. On the other screen they can see the user's face and work area, which is useful for predicting the modality that the user will use.

More details about adapting the WOz methodology for multimodal systems can be found in [11].

5. EXPERIMENTAL RESULTS

We have performed several experiments with the Archivus system. One of the aims of these experiments was to test and validate the above described multimodal dialogue management algorithms and WOz interfaces. Another one was to investigate if the language-related modalities (i.e. typed and spoken natural language input) are actually used, and how their use depends on the user and/or the task being performed. Answers to those questions provide an experimental validation (or invalidation) of a true need for multimodal systems, and corroborate their contribution to a more natural and flexible interaction. The collected data can also serve as a foundation for improving the parts of the system that are intensively used.

5.1 Experimental Setup

The experiments, which were carried out using the Wizard of Oz technique, involved ten volunteers (Archivus system users). Each experiment was divided into two twenty-minute sessions, during which the users used the Archivus system to solve tasks specified by the experimenters. The tasks included questions like *"Who attended all meetings?"*, *"What was the budget for the room furnishing?"*, *"Did they agree on a neutral colour scheme?"*, etc.

In the first session, only a subset of all the input modalities was available to the users, for example voice only, or pointing and typed natural language only. In the second session, which immediately followed the first one, the users were allowed to freely use any of the modalities available in the system (speech, typed natural language and mouse clicking). To familiarize users with the system and to intro-

duce the available modalities to them, each twenty-minute session was preceded by an approximately fifteen-minute tutorial explaining how to control each part of the system with the available modalities.

The statistical results presented in the next sections come only from the analysis of data obtained during the second sessions, after the users were familiar with the system and were allowed to use any modality combination they found natural and which satisfied their needs.

5.2 Use of Advising Prompts

As explained in Section 4.5, one of the goals of the experiments was to determine whether the prompt suggested by the dialogue system is appropriate in a given situation. The multimodal dialogue system contained 130 predefined prompts (340 when counting prompt reformulations) mainly of the "requesting" type. During the course of the dialogue, the output wizard was allowed to replace the system-suggested prompt by any other prompt (typically by an "advising" prompt) that was more suitable in the given dialogue situation.

During 200 minutes of interaction (10 users × 20 minutes session), the wizard used about 30 different prompts, however only 8 of them were used more than 4 times. The system suggested 830 prompts, of which 148 (i.e. 18%) were changed by the wizard. For example, when the user opened a meeting book, the default system prompt was *"Which part of the meeting would you like to access?"*. However, if the user had specified in the immediately previous interaction step that (s)he was looking for a document that was presented during that meeting, the wizard changed the default system prompt to something like *"You can find the document at the end of the book"*.

This confirms that a fully automated system (without a wizard) needs to be able to handle the "advising" prompts. However, a careful analysis of the experiment logs needs to be performed in order to identify relevant conditions and the variables that play a role. If a sufficient amount of data from the Wizard of Oz experiments is available, some machine learning methods could be applied for automated detection of the correct prompt during dialogue run-time.

5.3 Input Wizard Reaction Time

As already mentioned, the wizard's reaction time is perceived by users as the system reaction time. As this may in turn affect a user's choice of modality, the Input Wizard's reaction time should be constant and balanced over all modalities. Figure 5 shows the Input Wizard's average response times achieved during the experiments. While the reaction time needed to process spoken and typed input seems to be quite balanced (about 4–5 seconds), clicking is processed semi-automatically by the system (the wizard only needs to confirm it) and is therefore faster (1 second). This partially unbalanced system reaction time needs to be taken into account when considering user's modality choice.

To further balance the time difference, a possibility would be to degrade the speed of the pointing modality. However this is not advisable since most users expect pointing to be fast and may react negatively to the system as a whole.

5.4 Modality Use

One question that attracted our attention was how often the language-oriented modalities (i.e. spoken input and typed input) were used during interaction with the system. For every user, we counted the number of times they used speech, typed in a text request or command and the number of times they controlled the system by mouse. Language-oriented modalities were used for entering 31% of all user's inputs (22% speech, 9% typing), which is beyond a doubt an important portion of the interaction. This result can be therefore seen as an experimental justification of the effort directed from GUI-only towards natural language enabled systems.

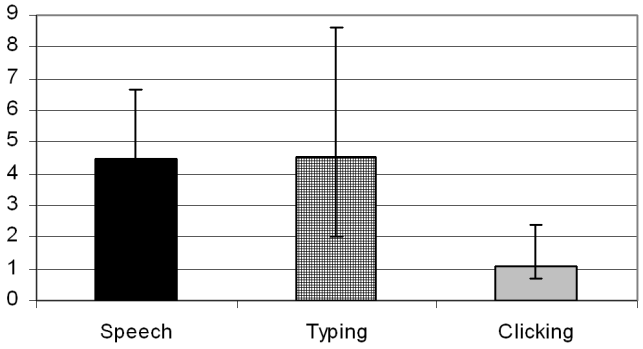


Figure 5: Average Input Wizard reaction times (in seconds) for each input modality.

Because all spoken input was processed by the wizard and not by an automated speech recognition (ASR) system, we achieved an almost perfect recognition rate which is unlikely to be achieved by current speech recognition technologies. The user's behaviour was therefore not affected by speech recognition errors and the users could really use speech whenever they found it more convenient than other modalities. In our system, use of speech was dominant for some actions. For example, speech was used in 54% cases for providing the search constraints, and taken together with typing accounted for 92% of the cases. It is very likely that use of speech would decrease in a system using ASR. However, the results suggest that speech and language-oriented modalities in general can be very useful when applied to appropriate tasks.

Our results also showed that mouse clicks are mainly used for controlling and navigating within the graphical interface. This may suggest that command-and-control desktop applications (i.e. applications that, besides providing a GUI, allow for vocal control of the GUI, without any sophisticated language processing and interaction) are not the appropriate domain for employing speech.

6. CONCLUSIONS

We have shown that a vocal dialogue manager, including a WOz assistance platform, can be adapted to control multimodal interaction with users. The extensions made to the dialogue strategy were mainly related to the fact that our multimodal system is equipped with a screen, which substantially changes the way the users want to interact with it. Users prefer to take more initiative in the communication and the system plays a more passive role compared to vocal-only dialogue systems. We assume that this is caused by the fact that the user has more information about the state of the interaction and current communication focus thanks to the presence of a screen. Consequently, a new

type of prompts appears to be necessary, one which gives context-dependent advice to the user.

The extensions made to the WOz interface were quite substantial since the multimodal WOz environment is significantly more complex. Due to high cognitive load, two wizards are required to supervise and control the system.

Even if the dialogue system equipped with a screen resembles traditional GUIs, the option of speech and natural language communication has been perceived as useful by the users. Our preliminary experiments showed that the use of natural language is preferred over using mouse for providing search constraints (i.e. entering data) to the system, while mouse pointing is more suitable for navigation within the graphical interface.

7. FUTURE WORK

While the dialogue strategies for selecting a new focus of conversation (i.e. another GDN) were relatively easy to modify for the multimodal settings, dialogue prompts are not satisfactory in all cases. So far, improper prompts were modified manually by a wizard during WOz experiments, but the dialogue model will have to be modified to include and properly use the new types of prompts.

Another issue that we should pay attention to is how the current multimodal interaction model will change in reaction to lower speech recognition performance. Current state-of-the-art speech recognition and language understanding does not reach the quality achieved during our WOz experiments. It would be interesting to identify the minimal required performance of those technologies which still allows a user to use speech conveniently for communication, similarly as it was done for vocal dialogue systems in [13]. This may lead to the introduction of new multimodal mechanisms for error handling in our dialogue model, for example a mechanism that allows the user to clarify the meaning of his last utterance by selecting from a small set of close possibilities.

8. ADDITIONAL AUTHORS

Additional authors: Marita Ailomaa (École Polytechnique Fédérale de Lausanne (EPFL), Artificial Intelligence Laboratory (LIA), email: `marita.ailomaa@epfl.ch`), Agnes Lisowska (University of Geneva, ISSCO/TIM/ETI, email: `agnes.lisowska@issco.unige.ch`) and Martin Rajman (École Polytechnique Fédérale de Lausanne (EPFL), Center for Global Computing (CGC), email: `martin.rajman@epfl.ch`).

9. REFERENCES

[1] D. Bohus and A. Rudnicky. Larri: A language-based maintenance and repair assistant. In W. Minker, D. Bühler, and L. Dybkjaer, editors, *Spoken Multimodal Human-Computer Dialogue in Mobile Environments*, volume 28 of *Text, Speech and Language Technology*, pages 203–218. Springer, 2005.

[2] T. H. Bui, M. Rajman, and M. Melichar. Rapid Dialogue Prototyping Methodology. In P. Sojka, I. Kopeček, and K. Pala, editors, *Proceedings of the 7th International Conference on Text, Speech and Dialogue—TSD 2004*, Lecture Notes in Artificial Intelligence LNCS/LNAI 3206, pages 579–586, Brno, Czech Republic, September 2004. Springer-Verlag.

[3] P. Cohen. The role of natural language in a Multimodal Interface. Technical Note 514, Computer Dialogue Laboratory, SRI International, 1991.

[4] N. Dahlbäck, A. Jönsson, and L. Ahrenberg. Wizard of Oz Studies – Why and How. In D. M. Wayne D. Gray, William Hefley, editor, *International Workshop on Intelligent User Interfaces 1993*, pages 193–200. ACM Press, 1993.

[5] A. Lisowska, M. Rajman, and T. H. Bui. ARCHIVUS: A System for Accessing the Content of Recorded Multimodal Meetings. In *In Procedings of the JOINT AMI/PASCAL/IM2/M4 Workshop on Multimodal Interaction and Related Machine Learning Algorithms, Bourlard H. & Bengio S., eds. (2004), LNCS, Springer-Verlag, Berlin.*, Martigny, Switzerland, June 2004.

[6] M. F. McTear. Spoken dialogue technology: Enabling the conversational user interface. *ACM Computing Surveys*, 34(1):90–169, 2002.

[7] S. Möller, J. Krebber, A. Raake, P. Smeele, M. Rajman, M. Melichar, V. Pallotta, G. Tsakou, B. Kladis, A. Vovos, J. Hoonhout, D. Schuchardt, N. Fakotakis, T. Ganchev, and I. Potamitis. INSPIRE: Evaluation of a Smart-Home System for Infotainment Management and Device Control. In *International Conference on Language Resources and Evaluation (LREC)*, volume 5, pages 1603–1606, Lisbon, Portugal, 2004.

[8] L. Nigay and J. Coutaz. A generic platform for addressing the multimodal challenge. In *CHI'95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 98–105. ACM Press/Addison-Wesley Publishing Co., 1995.

[9] S. Oviatt. Mutual disambiguation of recognition errors in a multimodel architecture. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 576–583, New York, NY, USA, 1999. ACM Press.

[10] N. Pfleger. Context based multimodal fusion. In *ICMI '04: Proceedings of the 6th international conference on Multimodal interfaces*, pages 265–272, New York, NY, USA, 2004. ACM Press.

[11] M. Rajman, M. Ailomaa, A. Lisowska, M. Melichar, and S. Armstrong. Extending the Wizard of Oz methodology for language-enabled multimodal systems. In *Proc. of the 5th International Conference on Language Resources and Evaluation (LREC)*, Genoa, Italy, May 2006.

[12] A. I. Rudnicky. Multimodal dialogue systems. In W. Minker, D. Bühler, and L. Dybkjaer, editors, *Spoken Multimodal Human-Computer Dialogue in Mobile Environments*, volume 28 of *Text, Speech and Language Technology*, pages 3–11. Springer, 2005.

[13] S. Schmidlin. Comparing mixed-initiative and system-driven approaches for dialogue design. Master's thesis, EPFL, 2003.