ARCHIVUS: A System for Accessing the Content of Recorded Multimodal Meetings

Agnes Lisowska[1], Martin Rajman[2], and Trung H. Bui[3]

[1] ISSCO/TIM/ETI, University of Geneva, 40 Bv. du Pont-d'Arve, 1211 Geneva, Switzerland
Agnes.Lisowska@issco.unige.ch
[2, 3] CGC/IC, LIA/IIF/IC, Swiss Federal Institute of Technology Lausanne,
Bat. INR, 1015 Lausanne, Switzerland
{Martin.Rajman, Trung.Bui}@epfl.ch

Abstract. This paper describes a multimodal dialogue driven system, ARCHIVUS, that allows users to access and retrieve the content of recorded and annotated multimodal meetings. We describe (1) a novel approach taken in designing the system given the relative inapplicability of standard user requirements elicitation methodologies, (2) the components of ARCHIVUS, and (3) the methodologies that we plan to use to evaluate the system.

1 Introduction

In this paper we describe a multimodal system, ARCHIVUS, which allows users to access and retrieve the content of stored multimodal meetings, either through directed search or browsing. Section 2 describes our particular approach to design, including the reasoning behind the choice of elements in the system and the initial user requirements on which the design is based. Section 3 explains the input and output modalities used in the system. Section 4 explains the interaction metaphor that we have chosen for ARCHIVUS, while section 5 provides a more detailed look at how the metaphor is implemented in the system. Section 6 describes the dialogue model that will be implemented, and the ARCHIVUS dialogue management strategy. Section 7 discusses the types of data the system is capable of accessing and the data that will be used in the early stages of system development for testing purposes. Section 8 discusses how we intend to evaluate and further refine the system, while section 9 provides a brief overview of work that is planned for the near future.

2 Approach to Design

ARCHIVUS was conceived with the intent of providing users with a convenient, intuitive and multimodal means to access processed and stored recorded multimodal meetings. Currently, records of meetings are stored primarily in text format as transcripts or meeting minutes. Only occasionally can audio, and even less frequently video, records be found. Moreover, rarely are these records stored in the unified, searchable and multimedia manner proposed in the multimodal meeting domain. This fact imposes significant limitations on how the data can currently be manipulated and

S. Bengio and H. Bourlard (Eds.): MLMI 2004, LNCS 3361, pp. 291 – 304, 2005.

reviewed. The multimodal meeting domain strips those limitations. However, one cannot assume that all of the wide range of modalities and media made available in this domain will be useful to real users in real life situations to the same extent. Consequently, the realistic needs of the users themselves must be carefully considered and accounted for in system design from the earliest stages.

While there are several known methodologies for designing software from the human computer interaction perspective [1], they all address cases where the way in which a human performs the task that the software is intended to facilitate or replace is already well known and well defined. We believe that for the multimodal meeting domain these methodologies are to a large extent simply not applicable. Potential users of a system such as ARCHIVUS have little or no prior experience with the types of tasks that the system avails. Thus, one of the first challenges was to determine how to gather a set of valid user requirements for a system enabling a task that is unfamiliar to the user.

The approach to requirements gathering that we have taken with the preliminary design of the ARCHIVUS system is driven by informed/grounded intuition. In a first attempt at developing user requirements for the multimodal meeting domain, IM2 [2] project members developed a set of natural language 'queries', or potential questions to the system. These 'queries' were meant to be indicative, at an abstract level, of the types of functionality that the system would need to account for. The results can be found at [3]. However, the query set was both potentially biased (due to the fact that all of the participants were directly involved in the project) and hard to analyze, due to a lack of a coherent set of guidelines for outlining the context.

In order to overcome these two problems, a second, more principled study was performed where the participants included both those involved and uninvolved in the IM2 project, and included individuals coming from a wider variety of backgrounds, ranging from physicians to administrators. Additionally, to facilitate analysis of the results, the second study was structured in such a way that each participant was asked to imagine themselves in one of four potential use cases – a manager tracking project progress, a manager tracking employee performance, a new employee needing to get background on a project, and an employee who has missed a meeting about a project that they were involved in. We believe that these four use cases, in addition to that of a person wanting to verify a specific fact, are likely to be the most common cases in which users might need the ARCHIVUS system and consequently, they provide a sound framework in which to situate our work. A detailed discussion of the results of this study can be found in [4]. We believe that the user requirements that can be extrapolated from careful analysis of the queries, coupled with standard design principles and our own intuitions, serve as a sufficient starting point for the design of the preliminary version of ARCHIVUS.

Finally, the platform and environment for which we are designing the first versions of the ARCHIVUS system is that of a laptop or desktop PC used in an office environment. This decision was based on two facts. The first is that a PC environment reduces learning curves introduced by new hardware. The second is that while the use of PDAs and mobile phones is rapidly gaining popularity we are not convinced that the general population is sufficiently familiar with their use for information browsing and retrieval, nor that the devices are powerful enough to handle the information

types in question. However, modification of the ARCHIVUS interface for use on handheld devices may be investigated in future work.

3 Modalities in ARCHIVUS

One of the key aims of ARCHIVUS is to provide flexible and consistent access to a variety of modalities in order to allow the user to interact with the system in the way that is most comfortable for them. This implies that all input modalities should be consistently and completely interchangeable. The user should always be able to use any of the modalities in any context and have the option of naturally changing between them if they find that communication using one of the modalities is not yielding the required results.

In the preliminary version of the system we are planning the inclusion of three active input modalities, one passive input modality, and three output modalities. The three active input modalities are voice, pointing (either through a mouse or a touch screen) and text (keyboard). The passive modality is emotion recognition, which will be used to detect the emotional state of the user and modify the dialogue strategy accordingly. The output modalities will include graphics, sound and text.

When incorporating the results of automatic speech recognition (ASR) into a system (in our case the speech input), the robustness of that system in the face of common problems encountered with ASR is called into question. In our work, we assume that the data being accessed, even if has been gathered using ASR, will have been checked and corrected by humans (at least until ASR technology provides adequate results). However, robustness of the system in the face of real ASR results can be tested using a noise generator, which simulates varying degrees of noise in text. Using the generator, we can test acceptability thresholds for ASR errors given the availability of different modality combinations. Finally, we believe that the simultaneous availability of various input modalities will also allow the user to overcome ASR problems by allowing them to actively supplement the voice modality with other modalities.

4 The ARCHIVUS Metaphor

Interaction metaphors are often used in interface design to explain to a user, in terms and concepts with which they are already familiar, how to interact with an unfamiliar application and what its functionalities might be [1]. As explained in section 2, many of the specific functionalities offered by the ARCHIVUS system will be unfamiliar to users, so it was vital to find a suitable metaphor in which to situate the system.

The metaphor that we have chosen is that of a person using an archive or library to find the information they are looking for. We believe that this metaphor will help users naturally discover and exploit the full capabilities of the system. We have found that meetings can be mapped quite easily and naturally to the content of a standard book and that a database of meetings can be mapped to the structure of a library. Moreover, we believe that the metaphor can be extended to sufficiently cover all foreseen interface functionalities.

Thus, in ARCHIVUS, the database of stored meetings is represented by a series of bookcases, each of which contains representations of meetings that are available in the database. Each individual meeting is represented as a book, and a set of related meetings as volumes of a series. The title of the book becomes the subject of a meeting, while the authors are the meeting participants. The publisher's information page can contain the time, date and location of the meeting, as well as the institutes that were involved. The table of contents corresponds to the agenda of the meeting, listing the topics of the meeting. Chapters in the book always represent individual topics in a meeting, chapter sections reflect the dialogue structure of the topic, and individual paragraphs are the specific utterances that participants make. The appendix represents a list of the documents that were used or referred to during the meeting, while the index is a list of keywords from the meeting.

Finally, while the user is interacting with the system, they will have access to an electronic notebook into which they can copy relevant sections of a meeting for future reference, or make online notes, just like the users of a regular library might have a notebook in which to write down their findings.

5 The ARCHIVUS Interface

The ARCHIVUS system has been designed as a hybrid search and browsing system. We feel that this combination of interaction paradigms is the best solution to suit the user needs for this domain and maximize the flexibility of the system in terms of the types of interactions that it allows and the level of detail of information that becomes available to the user. Moreover, searching in ARCHIVUS can be done in one of two ways. The first is through constraint satisfaction, where the user imposes constraints on the search space until the latter is sufficiently reduced to provide a single solution, or in the worst case, a small set of solutions. The other search approach uses linguistic analysis of a natural language question posed to the system to drive a keyword based search and return relevant results.

The results of either of these searches, and in many cases of the browsing as well, can be viewed on several levels. The highest, global, level is represented on the bookcase. In this case, the search has been underspecified and there remain a large number of meetings that meet the criteria. These meetings are active and visible on the bookcase. The second, local, level is the case where only a single meeting matches the search criteria. In this case, the relevant meeting is shown in the interaction area. The final case is at the target-specific level, where the search results yield a particular section of a meeting. Here, the result is that the relevant section of the meeting is shown directly.

Furthermore, while we consider our system to be dialogue based, it is not dialogue in the strictly traditional sense of the word. In the ARCHIVUS system, dialogue can also refer to an 'interactive dialogue' – a set of interactions between the user and the system that are not necessarily rooted in speech. This distinction is important since in ARCHIVUS, the user has constant access to any of the three primary modalities (voice, text and pointing) and can use any of them in any combination, or interchangeably, without affecting the interactive dialogue.

The user interface itself can be divided into seven general areas, show graphically in figure 1 and explained in detail in the following sections.

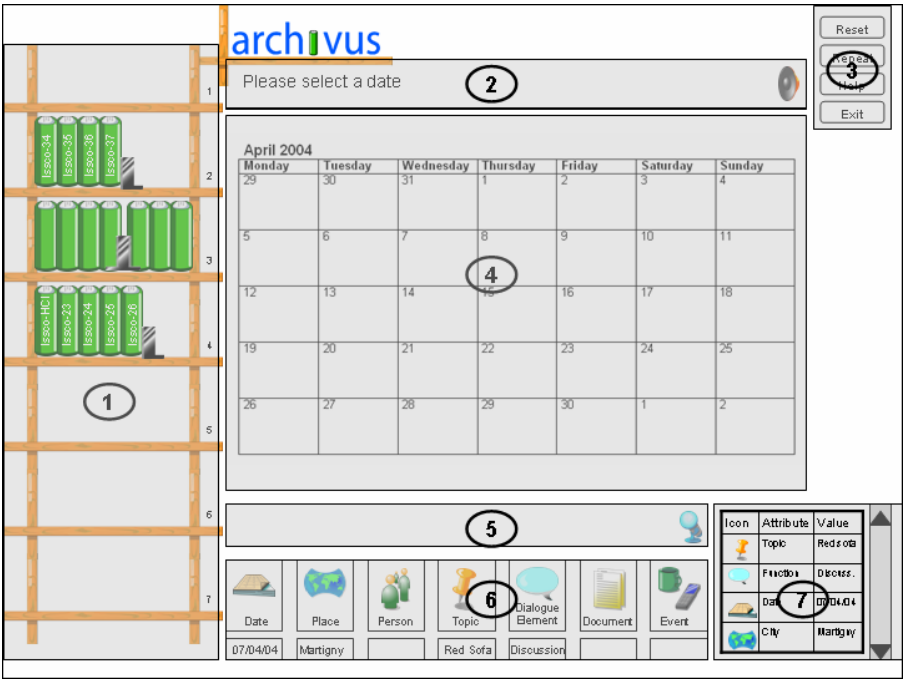

Fig. 1. The overall interface

5.1 Bookcase

The bookcase area serves two primary purposes. The first is to provide an overview of the entire database to the user. All of the meetings in the database will be represented as books on the bookcase. This global database representation gives the user constant and active feedback of the effects of their searches and criteria on the database, which can guide them in further decision-making and refinement steps. The feedback is provided through the use of shading, where brighter coloured books are those that are active (still fit the search criteria), while those that are dimmed are inactive (no longer fit the search criteria). Finally, books that are related are grouped together whenever possible, and are separated from other books by bookends.

The second purpose of the bookcase is to allow the user to browse the database without having to interact with the search/dialogue engine. It should be kept in mind that browsing can be done using any of the available modalities, including voice. When browsing, the user can select any of the meetings at any time, in which case they open in the interactive display area (5.4). Furthermore, the user can sort the meetings by changing the labels on the legs and shelves of the bookcase. The specific default settings for bookcase labels and the exact nature of the user's interaction with the bookcase will be determined over the course of the user-centered evaluation experiments described in section 8.

5.2 Prompt Bar

The prompt bar is used to visually display all prompts from the system. A "speaker" icon on the right-hand side indicates the availability of speech output.

5.3 Function Buttons

In the upper right-hand corner of the interface there are four function buttons – three directly related to the dialogue mode (*Reset, Help,* and *Repeat*), and one system button (*Exit*). When the *Reset* button is selected, all of the constraints that have been accumulated during a dialogue are erased and the system is reset to accept a new query. The *Repeat* button enables the user to hear the last prompt again, in cases where they misheard or misunderstood it. The *Help* button provides access to online help which explains the functionalities available in the interface, while the *Exit* button quits the ARCHIVUS system altogether.

5.4 Interactive Display Area

This area serves three purposes depending on the point of the interaction at which the user finds themselves; a visual display of the interactive dialogue mode, a space in which to view a meeting book, and a space in which to view multimedia elements of the database such as video and accompanying documents.

5.4.1 Interactive Dialogue Mode

When the user selects one of the criteria refinement buttons (see 5.6), the interactive dialogue mode is launched and a visual representation of the current state of the dialogue appears in the interactive display area. For example, a calendar showing possible meeting dates is displayed when the *Date* button is selected (see Fig. 1).

5.4.2 Viewing a Meeting Book

When the user has reached the book level, we expect that they will primarily be browsing the meeting. As was mentioned in section 4, a book is the representation of the meeting and consequently all aspects of the meeting should be accessible by interacting with the book. Figure 2 shows a meeting book in the ARCHIVUS system. The top line indicates the name of the meeting and the current chapter being viewed. The utterances of each participant are colour-coded to facilitate browsing. Notes in the margin indicate the name of the speaker of an utterance (much as in a play), and the presence of any accompanying documents related to an utterance or set of utterances is indicated by the document icon. The user can move through the pages of the book using the next/previous arrow buttons.

To either hear the original audio soundtrack from a part of the meeting or view the video, the user selects the appropriate audio/video icon at the bottom of the page, and then selects the utterance that they wish to hear/see. The audio/video playback will start at the selected utterance and continue until the user explicitly stops the playback.

Tabs are used to allow the user to jump to different sections of the book. The tabs on the right-hand side of the book provide links to the constant elements found in all books such as the cover, the table of contents, the index and the appendix. The tabs on the left-hand side of the book change dynamically and represent the individual 'hits'

in the book that fit the current search criteria. Each of these tabs is numbered in x/y format where x is the number of the hit and y is the total number of hits in the book.

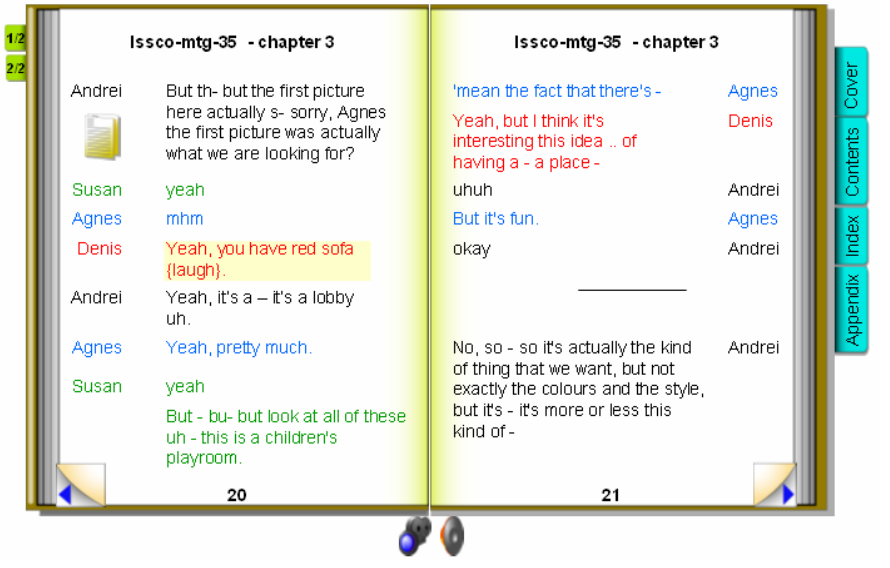

Fig. 2. An ARCHIVUS 'meeting'

Additionally, within a page, the section that is most relevant to a particular query will be 'highlighted'. Thus, if the query asks to find a discussion about some topic, the entire discussion will be highlighted, where this highlight can span several pages if the discussion happens to be long. This mechanism also allows for high level browsing of the book if the user wishes to view information at a more abstract level than that of individual utterances. The table of contents, index and appendix will be reminiscent of a web-page, where entries such as chapters, keywords and documents are clickable, and clicking on them automatically takes the user to the relevant part of the book, or opens the relevant document in the viewing area.

5.4.3 Viewing Video, Audio and Documents

The viewing area also serves to display multimedia such as video and electronic documents. Standard VCR style buttons will be used to control video and audio, while traversing documents will follow a web-like style. The viewing area, in relation to the examination of particular documents, will also allow for the incorporation of work being done by the IM2.DI group at the University of Fribourg, where they are working on interactively linking segments of documents to the part of the meeting where those segments were discussed [5]. Consequently, interacting with a document should affect the view of the book that the user has. For example, clicking on another section of the document might take the user to a different page in the book from the one from which the document was initially launched.

5.5 User Input Box

The user input box is where the user can either type an information request, or, in the case of speech input, the results of the speech recognition software appear. This area effectively allows the user to initiate a natural language driven directed search, where the user already has some idea of the information that they are seeking and poses a specific request for information to the system. Additionally, the microphone icon on the right side of the box indicates the availability of voice as an input modality.

5.6 Criteria Refinement Buttons

In order to help refine a request during the dialogue mode the interface is equipped with criteria refinement buttons. These buttons, selectable using any of the three input modalities, help the user refine a particular request. There are seven criteria refinement buttons planned for the current version of ARCHIVUS. *Place* allows the user to specify the location of a meeting. *People* shows the user the possible attributes of people that they can search on such as first name, family name, function, affiliation or photo. *Date,* allows the user to search by year, month, day and am/pm. *Topic* shows the user possible topics to search for, and indicates the keywords and meetings associated with the topic. *Dialogue Elements*, allows the user to launch a search for specific aspects of a dialogue such as discussion, argumentation, questions, decision making etc. *Events*, allows the user to search for events such as people leaving the room, mobile phones ringing, people drinking etc. *Documents* allows the user to specify the type of document to search for such as presentation slides, notes, whiteboard activity, and electronic versions of paper artifacts from the meeting.

5.7 History Area

A well-known HCI design principle is that the user should always be able to follow and backtrack along the path which they used to reach a particular point in their interaction. To this end we provide the history area, which shows the user, in iconified and textual form, the constraints that they have imposed in order to reach their goal. The content of the history will be represented in a scrollable pane in reverse chronological order.

6 Dialogue System

The multimodal dialogue system for ARCHIVUS will be based on the Rapid Dialogue Prototyping Methodology (RDPM) [6] developed at the EPFL's Artificial Intelligence Laboratory. The general idea underlying the proposed RDPM is that the dialogue model is a finite-state model that can be quite easily and systematically derived from a relational representation of the application itself, hereafter called the *task model*. More precisely, the RPDM consists of five main consecutive steps: (1) *producing a task model* for the targeted application; (2) automatically *deriving an initial dialogue model (and the associated multimodal dialogue-driven interface)* from the produced task model; (3) using the generated interface to *carry out Wizard-of-Oz experiments* (i.e. dialogue simulations) to improve the initial dialogue model; (4) *carrying out an internal field test* to further refine the dialogue model

(reformulation of system messages, improved feedback, etc.), and validate the evaluation procedure (coherence, understandability); and (5) *carrying out an external field test* to evaluate the final dialogue model. In the following sub-sections, steps 1 and 2 will be described in the ARCHIVUS context. Steps 3, 4, and 5 are described in [6] and section 8.

6.1 Producing the Task Model

In the RDPM, an application is seen as a set of functions the user can invoke through the multimodal interface to perform the various functionalities provided by the application. In this perspective, an application is modeled as a set of relational tables, where the rows correspond to the possible functions (also called "solutions" or "targets") and the columns are the attributes needed to uniquely identify each of the functions, and to invoke it.

In other words, the values of the attributes in a row of the solution table (also referred to as *canonical values*) correspond to the values of the arguments of the function, the call of which results in the fulfillment of the corresponding application functionality. For example, in the IM2 domain the task model can reduce to a single generic function `select_meeting(name, place, participants,…)`, the attributes of which identify the selection features available for the meeting search. Therefore, the task model of the IM2 project is simply a table with as many columns as there are attributes, the rows of which are the various value combinations corresponding to existing meetings. At the computational level, the calls to the `select_meeting()` function are implemented in the form of SQL queries to the project database containing the required information.

Notice that the current version of the RDPM presupposes that the task model consists of a single relational table, also called the *solution table*. However, in the case of more complex models consisting of several interconnected tables (for example a main table containing the acceptable value combinations of the main attributes and several additional tables relating the values present in the main table to additional attributes), standard database normalization procedures (such as join operations) can first be applied to transform the original tables into a single (eventually large) one.

6.2 Dialogue Model

In our approach, a dialogue model is defined as a set of interconnected multimodal Generic Dialogue Nodes (referred to as mGDNs, [i.e. 7]), where each of the dialogue nodes is associated with one of the attributes (also called “slots” or “fields”) in the solution table. In complex applications such as ARCHIVUS, these mGDNs are divided into groups, where each group element is considered as an object and the mGDNs in the group are attributes of the object. For example, the *First Name*, *Last Name*, and *Function* mGDNs belong to the *Person* group. For any given slot, the role of the associated mGDN is to perform the simple interaction with the user that is required to obtain a valid value for the associated attribute.

In the architecture that we have selected for the implementation of our multimodal dialogue-driven interfaces, the processing of the mGDNs (i.e. the actual interaction with the user according to the specification of the mGDNs) is performed by a specific module called the *local dialogue manager*. However, this is not sufficient to carry out any real dialogue: some form of global dialogue management also has to be

integrated. For example, in addition to the definition of the mGDNs and the specification of the local dialogue manager, some branching logic responsible for the management of the global dialogue flow needs to be specified. In our approach, this branching logic is hard-coded in a specific dialogue management module, called the *global dialogue manager*. The underlying assumption is that the encoded local and global dialogue flow management strategies are indeed application-independent, i.e. that, in most situations, they lead to an acceptable, though not always optimal behavior for the system. Consequently, in our approach, dialogue model design essentially reduces to the application-dependent, declarative specification of the mGDNs, the encoded dialogue management strategies being used without modification for all applications.

6.3 Multimodal GENERIC Dialogue Nodes – mGDNs

Each mGDN is designed to support three active modes - text, voice, and gesture - which are used either simultaneously or independently depending on the configuration of the mGDN. The output of an mGDN consists of semantic pairs of the form (attribute name, attribute value). Therefore, a fusion mechanism (i.e. local fusion module) needs to be used inside each mGDN to combine the specific output produced by each of the input modalities and to produce the validated output semantic pairs.

To deal with the various attributes appearing in the solution table defining the task model, we consider three main types of mGDNs: (1) Simple mGDNs (also called static mGDNs) associated with *static fields*, (2) List Processing mGDNs (also called dynamic mGDNs) associated with *dynamic fields,* and (3) Internal mGDNs, used to perform the interactions that are required by various special functions implemented in the dialogue manager (e.g. confirm a selected solution, start/reset the dialogue, etc.).

As already mentioned, the role of each mGDN is to perform a simple interaction with the user to obtain a valid value for the associated attribute. In this perspective, the difference between static and dynamic mGDNs is that the former are expecting the user to directly provide a value for the associated attribute. For example, a static mGDN associated with the "Location" attribute might ask a question such as "*What location are you looking for?*" and will be expecting an answer containing a value taken from a predefined list of values such as "*Lausanne*", "*Geneva*" or "*Martigny*". On the other hand, a dynamic mGDN will ask the user to choose from a dynamically computed list of values. For example, a dynamic mGDN associated with the "Topics" attribute will generate a list of topics and ask the user to indicate the position of their selection in the list. The List Processing mGDNs are an important component of the targeted dialogue model as they allow efficiently taking into account dynamic vocabularies that could not be reliably processed by simple mGDNs because of the limited performance of the speech recognition module in such conditions. To realize the interaction for which it is responsible, each mGDN contains two main types of components: multimedia prompts and grammars.

Multimedia prompts: Multimedia prompts contain messages and a pointing zone. The messages are visualized in the user interface and/or uttered by the mGDN during the interaction and are combined with a pointing zone (the content of which is a map, calendar or table depending on the nature of the mGDN) to allow the user to provide the desired values using keyboard, microphone or mouse click/touch screen. Several types of prompts are defined, among which are the main prompt,

corresponding to the initial question asked by the mGDN, and the help prompt that is uttered/visualized if a request for help is expressed by the user. The formulation of the prompts plays an important role during the dialogue. In particular, it influences the level of *mixed initiative* (i.e. the degree of flexibility that the system allows in the interaction). For instance, a main prompt such as "*What are you looking for?*" expresses the fact that the system is ready to accept a broad range of user requests, while a more precise prompt such as "*Do you want to select a meeting, yes or no?*" implies a low level of mixed initiative as the user is only expected to answer either *yes* or *no*.

Grammars: The role of the grammars is to make the connection between the surface forms in the natural language utterances made by the users and the canonical values used in the task model (the set of values defined for the attributes associated with the mGDNs in the solution table for the application). As such, the grammars represent the main Natural Language Processing elements in the system, and might also be used in the speech recognition engine to improve recognition quality.

Finally, each mGDN is always associated with some specific global grammars such as the help and repetition grammars. These grammars correspond to *Request for Help* and *Request for Repetition* situations which are mentioned in the next section.

6.4 Local Dialogue Flow Management Strategy

Each mGDN is able to locally process five types of generic situations: (1) *OK*: the user answers the question in an acceptable way; (2) *Request for Repetition*: the user asks for repetition of the last system prompt; (3) *Request for Help*: the user does not know how to answer the question and asks for an explanation; (4) *NoInput*: the user neither produces an utterance nor uses other input modalities; and (5) *NoMatch*: the user answers but nothing useful can be extracted from any of the input modalities.

In the case of the *OK* situation, control is simply handed back to the global dialogue manager which applies the global dialogue management strategy for the activation of the next mGDN. In the other four situations, control remains at the mGDN level. In these "problematic" cases, there is, therefore, a need to repair the dialogue and the system operates in the following way: (a) *Request for Repetition*: the current mGDN is reactivated and its main prompt is played if it is the first request for repetition, otherwise a reformulated prompt is played; (b) *Request for Help*: the mGDN is reactivated and the associated help prompt is played instead of the main prompt; and (c) *NoInput/NoMatch*: the current mGDN is reactivated and the NoInput/NoMatch prompt is concatenated at the beginning of the main prompt.

6.5 Global Dialogue Flow Management Strategy

Global Dialogue Flow Management (GDFM) consists of several complementary strategies:

- a branching strategy (also called *branching logic*) defining the next mGDN to be activated;
- a dialogue dead-end management strategy to deal with dialogue situations where no solution corresponds to the request expressed by the user;

- a confirmation strategy to provide the system with validation possibilities for the values acquired during the interaction;
- a dialogue termination strategy to define when the interaction with the user should be terminated (i.e. a solution proposed); and
- a strategy to deal with incoherencies (i.e. situations where there are at least two incompatible values provided by the user).

As already mentioned, all these strategies are encoded in the global dialogue manager and are, therefore, application-independent. These strategies are explained in more detail in [6]. Additionally, our dialogue manager can work either in system-driven or mixed initiative mode. We therefore use the information from the passive modality (i.e. emotion recognition) to automatically and smoothly alternate between the two modes to enhance natural communication between the system and the user.

7 ARCHIVUS Data

The data that the ARCHIVUS system will access is expected to be stored audio, video, electronic copies of all documents used in the meetings (including presentation slides, paper artifacts, whiteboard activity and participants notes), as well as annotated transcripts of the meetings, stored in XML format. In the preliminary stages of system development and user testing, the data that we will be using is a set of four meetings recorded by members of the IM2.MDM IP [8] in the IDIAP Smart Meeting room [9]. The topic, spanning all four meetings, is the furnishing of a lounge/reading room for use by the staff of a university research group. This topic, or scenario, was chosen with the particular view of performing user-centered evaluation of the system as it is easy to relate to for most users, which will help them maintain interest in the content of the meeting. This is important in order to ensure that the evaluation reflects a true evaluation of the system and not an evaluation biased by the interestingness of the data accessed. For similar reasons, the selected meetings have been kept natural in the interactions that they include, but have at the same time been refereed in such a way that factors that might affect the evaluators' cognitive involvement in the task, such as excessive cross-talk, have been avoided.

This data will be transcribed and annotated manually. The annotations will be based on a combination of tags outlined in the MALTUS tagset [10], and additional meta-tags, indicating non-dialogue or non-textual events, such as when someone left the room, which will be determined based on the types of meta-information that were requested in the query set gathered during the user requirements elicitation study described in section 2.

8 Evaluation

As with the design and development of most software and interfaces, our evaluation process will be highly iterative. In our case this is particularly necessary since, as mentioned, determining a priori user requirements poses difficulties. Consequently, the evaluation stage will serve two purposes; 1) evaluation of the system in the

traditional sense of evaluating its performance and ensuring that users can interact smoothly with it, and 2) the continued gathering and refinement of user requirements.

Given the uncertainty of the validity of the user requirements that we have gathered and the fact that the implementation of all levels of desired flexibility in the software and interface would be extremely costly, we have decided to adopt the Wizard of Oz (WOz) experiment methodology to our domain. The WOz methodology [11, 12], originally developed in the natural language processing community, is based on the notion that a user interacts with a system that has not been fully implemented. The user, however, is not aware of this fact and believes that they are interacting with a fully functional system. In reality, a human controller, known as a wizard, fills in the parts of the system that are missing.

This methodology allows both of the aforementioned evaluation goals to be met simultaneously. A user interacting with the system will be actively testing the already functional components of the system, thus satisfying the standard system evaluation aspect. At the same time, having the wizard supplement additional functionality allows the testing of design theories without having to implement them, and allows for the addition of functionalities 'on the fly'. Allowing for unforeseen actions lets developers explore new avenues of design and their feasibility in the system almost immediately, both in terms of integration into the system, and usefulness to the user. New functionalities and queries from each iteration will be integrated into the existing set of user requirements, and considered in future stages of refinement and development of the system.

Once a stable and complete ARCHIVUS system is developed, we hope to compare the performance of ARCHIVUS to other interfaces and browsers designed for the same domain, such as the Ferret Meeting Browser [13] being developed at IDIAP. In order to perform such a comparative test we plan to use the Browser Evaluation Test scheme proposed in [14].

9 Future Work

Development of the ARCHIVUS system is currently in progress. Four modules of the user interface have been partially completed (input from the keyboard, dialogue output prompt, bookcase, and pointing zone). The data described in section 8 has been transcribed and is now being annotated in detail. The immediate next steps will be completion of the implementation of the system, the development of a suitable Wizard of Oz testing environment, and an initial round of evaluation.

Acknowledgements

We would like to thank the Swiss National Science Foundation NCCR, the University of Geneva and the Ecole Polytechnique Federal de Lausanne (EPFL) for their support in funding this work, as well as Marcin Bogobowicz for his graphic arts skills.

References

1. Dix, A., J. Finlay, G. Abowd and R. Beale, *Human Computer Interaction* Second Edition, Prentice Hall, England, 1998.
2. IM2 webpage http://www.im2.ch/e/home.html
3. IM2.MDM webpage http://issco-www.unige.ch/projects/im2/mdm/
4. A. Lisowska, "Multimodal Interface Design for the Multimodal Meeting Domain: Preliminary Indications from a Query Analysis Study", Report IM2.MDM-11, Nov. 2003.
5. D. Mekhaldi, D. Lalanne and R. Ingold. "Thematic Alignment of recorded speech with documents", DocEng 2003, ACM Symposium on Document Engineering, Grenoble, 2003.
6. T. H. Bui and M. Rajman "Rapid Dialogue Prototyping Methodology", Technical Report No. 200401, Swiss Federal Institute of Technology, Lausanne (Switzerland), January, 2004.
7. E. Bilange, *Dialogue personne-machine, modélisation et réalisation informatique,* Langue, Raisonnement, Calcul, Hermès, Paris, France, 1992.
8. IM2 newsletter, May 2004.
9. D. Moore, "The IDIAP Smart Meeting Room", IDIAP-Com 02-07, 2002
10. A. Popescu-Belis, "Dialogue act tagsets for meeting understanding: an abstraction based on the DAMSL, Switchboard and ICSI-MR tagsets", Report IM2.MDM-09, September 2003.
11. N. Dahlbäck, A. Jönsson and L. Ahrenberg, "Wizard of Oz Studies – Why and How", in W.D. Gray, W.E. Helfley and Murray, D. (eds). Proceedings of the 1993 Workshop on Intelligent User Interfaces (pp. 193/200) Orlando, FL. New York, ACM Press, 1993.
12. D. Salber, and J Coutaz, "Applying the Wizard of Oz technique to the study of Multimodal Systems", 3rd International Conference EWHCI'93, East/West Human Computer Interaction, Moscow. L. Bass, J. Gornostaev, C. Unger Eds. Springer Verlag Publ. Lecture notes in Computer Science, Vol. 73. pp. 219-230. 1993.
13. Ferret Meeting Browser http://rhonedata.idiap.ch/documentation/Ferret_User_Guide/help.html
14. M. Flynn and P. Wellner, "In Search of a Good BET", IDIAP-Com 03-11, 2003.