Extending the Wizard of Oz Methodology for Language-enabled Multimodal Systems

Martin Rajman*, Marita Ailomaa*, Agnes Lisowska†, Miroslav Melichar*, Susan Armstrong†

*Artificial Intelligence Laboratory (LIA), École Polytechnique Fédérale de Lausanne (EPFL),
CH-1015 Lausanne, Switzerland
{martin.rajman, marita.ailomaa, miroslav.melichar}@epfl.ch
†ISSCO/TIM/ETI, University of Geneva,
CH-1211 Geneva, Switzerland
{agnes.lisowska, susan.armstrong}@issco.unige.ch

Abstract

In this paper we present a proposal for extending the standard Wizard of Oz experimental methodology to language-enabled multimodal systems. We first discuss how Wizard of Oz experiments involving multimodal systems differ from those involving voice-only systems. We then go on to discuss the Extended Wizard of Oz methodology and the Wizard of Oz testing environment and protocol that we have developed. We then describe an example of applying this methodology to Archivus, a multimodal system for multimedia meeting retrieval and browsing. We focus in particular on the tools that the wizards would need to successfully and efficiently perform their tasks in a multimodal context. We conclude with some general comments about which questions need to be addressed when developing and using the Wizard of Oz methodology for testing multimodal systems.

1. Introduction

In the traditional software design cycle there are three general stages – (1) user requirements gathering, (2) design and prototyping, and (3) evaluation – often executed in a cyclic fashion. In regular desktop-environment systems, although the graphical interface is dependent on both the input modes available (usually mouse and keyboard) and the functionalities that the system provides, the actual evaluation of the functionalities and the graphical user interface (and input modes) can to a large extent be done separately, particularly at the early stages of development. In multimodal systems however, and especially in systems that are designed for a domain that is relatively unfamiliar to the standard user such as accessing recorded multimedia meetings, this procedure becomes much more complex. Notably, it is much harder to do accurate requirements analysis, since users find it much harder to specify what tasks they would be performing and they cannot easily be observed in the foreseen environment. Furthermore, it might be unwise to set *a priori* assumptions about the types of things that users would want to do in a new domain and in particular how they would want to do them. In order to resolve these problems we feel that multimodal systems for unfamiliar domains should be developed in an environment that allows for simultaneous design/prototyping, requirements gathering and evaluation. Approaches used for either graphical mouse and keyboard interfaces or natural language-only interfaces alone seem insufficient for this.

Natural language interfaces are often developed and evaluated within the Wizard of Oz (WOz) methodology, see for instance (Lyons et al., 2005; Cheng et al., 2004; Geutner et al., 2002). In this methodology, a human 'wizard' simulates those parts of an interactive system that are not yet implemented or working optimally, with the aim of discovering the interaction model for a given application and/or testing all parts (modules) of the system within the context of the whole system. This also allows for both on-line processing of ambiguous input and the gathering of realistic data without heavy investment in implementation. The gathered data can then be used in the development of language models and dialogue strategies for the system. Conversely, doing non-WOz evaluations of natural language interfaces at early stages of design is impractical because it forces the implementation of unvalidated language models which is both time consuming and does not allow for processing of unforeseen input. The consequence of the former is that the user reduces their linguistic behaviour to what is covered by the existing language model rather than using language as they think would be useful in the context of the system or task (Dahlbäck et al., 1993).

We therefore propose a hybrid approach which allows for user-centered evolutionary system design and evaluation by extending the WOz methodology for multimodal use by giving the wizard control over both the natural language and graphical components of the system.

2. Wizard of Oz experiments for vocal dialogue systems

In order to highlight the differences between the design and evaluation of voice-only systems versus multimodal systems, we will first describe the particularities of WOz experiments for voice-only interaction, since vocal dialogue systems can be seen as a special case of multimodal dialogue systems, i.e. systems that allow interaction using only one modality (speech).

2.1. Supporting Voice-only WOz experiments

In earlier work we developed a methodology for the rapid design of vocal dialogue systems with mixed initiative (Bui et al., 2004). Using this methodology, the designer of the interactive vocal system only needs to provide a model of the application and the application-specific language resources (grammars, prompts). Our Java implementation

takes those resources and makes the dialogue system operational; it also automatically provides a graphical interface for the wizard which can be used to control or simulate some of the system functionalities.

In our experience, the modules that need to be simulated or supervised at the early stages are the Speech Recognition Engine (SRE), the Natural Language Understanding (NLU) module, and sometimes the decisions of the Dialogue Manager (DM). WOz experiments allow for the simulation of full grammar coverage (with an excellent speech recognition rate) and can also be used to find the minimal speech recognition performance necessary for smooth interaction flow (Krebber et al., 2004).

2.2. Representing and processing vocal input

In the implementation of our methodology, the user's vocal input is represented as a set of semantic pairs. A Semantic Pair (SP) is a qualified piece of information that the dialogue system is able to understand. For example, a system could understand semantic pairs such as `date:monday`, or `list:next` (the concrete set of SPs depends on the application). The wizard's interface allows them to produce a set of semantic pairs after every user's utterance (e.g. "*What are the other values in the list?*" could translate to `list:next`). When NLU is integrated into the system and SPs are automatically produced by the module, the wizard either confirms these generated SPs or modifies them if needed. In the case where the NLU works satisfactorily the wizard only confirms semantic pairs or simulates the output of the SRE (a transcription of the user's utterance).

2.3. Particularities of WOz experiments with vocal dialogue systems

We have found that WOz experiments in voice-only settings are relatively uncomplicated from the technical point of view. This is due to the fact that the dialogue manager and the wizard's interface are running as one application on one single computer (fully under the wizard's control) and only audio signals need to be transmitted to the user (test subject), making hardware and software setup relatively simple.

The cognitive load of the wizard is also lower in this case in comparison to WOz experiments with a multimodal dialogue system, because the wizard chooses his actions based mainly on listening to the ongoing dialogue, i.e. auditory input. In multimodal systems where a graphical interface is also available to the user, the wizard must also be aware of what is happening on the screen, which increases their load.

We also found that users tolerate the wizard's reaction time (which constitutes a large part of the overall system response time) if it is within a few (approx 5) seconds, since they are in generally not yet familiar with speech interfaces and seem to understand that processing of speech takes time. However, users will not accept slow response times with GUI-equipped systems, since they are accustomed to fast reaction times with such systems.

Additionally, the dialogue flow (dialogue states, prompts) usually does not need to be changed by the wizard for several reasons: (1) the dialogue flow is controlled by the system through mixed initiative – the user can provide more information, but the system decides what to do next, as opposed to GUI systems where it is primarily the user who decides where to go next, (2) the user's response is influenced only by the system prompts and not by the information on the system screen (which usually displays multiple pieces of information at the same time), (3) the task to be accomplished by voice-only systems is usually simpler than the tasks to be solved by systems equipped with a screen.

3. Extending the WOz methodology for multimodal systems

In order to be able to design a dialogue based multimodal system, we must look at four elements simultaneously – the dialogue management, the language model, the multimodal interaction and the graphical interface – each of which is directly influenced by user needs. These elements also influence one another, making their decoupling during the design and evaluation process impossible. For example, the sizes of the graphical elements might depend on whether touch is being used as an input modality, the choice of graphical elements depends on the dialogue management elements which in turn constrain the types of linguistic interactions possible, and these in turn play a more general role in influencing which modalities a user will choose to use for a specific task. (See Figure 1 for the types of influences and the scope of the traditional Wizard of Oz vs. the proposed extended methodology).

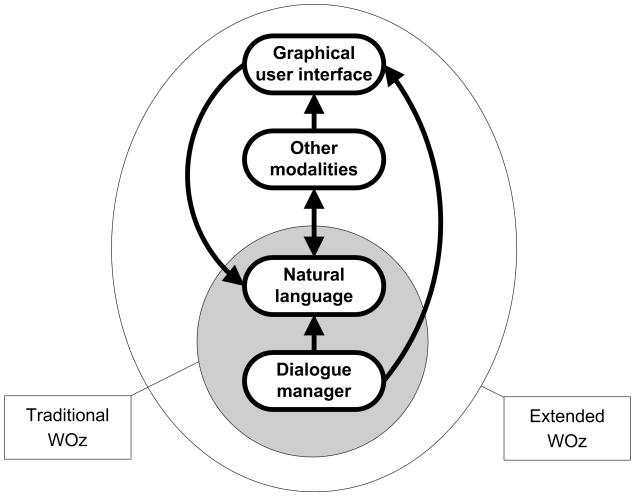

Figure 1: Influences of elements and scope of the WOz methodologies.

In terms of the wizard's simulation tasks there are several differences between controlling a unimodal language interface and a multimodal system. First, the wizard has to manually interpret or confirm generated interpretations from inputs in all modalities (for instance pointing, speaking and typing). If this is to be done in a uniform way, the wizard needs to have the same interpretation formalism for all modalities. Moreover, the degree to which modules for the different modalities are automated is not the same in all phases of system development, since the modules for some modalities are easier to automate than others. For instance,

pointing is easy to automate because the interpretation of a click is unambiguous. On the other hand, interpretation of speech requires a sufficient-quality speech recognizer and appropriate NLU algorithms, which means that the wizard will produce the interpretations manually for this modality in the initial phases of system development. The same is true for keyboard input, which is a modality that is not used at all in voice-only systems. Consequently, multimodal systems that allow for both voice and keyboard input automatically require extra development steps.

Another particularity of multimodal systems is that the wizard may need to control the dialogue flow more often than for a purely vocal system. This is because vocal systems don't immediately reveal all the functionalities to the user, whereas multimodal systems are equipped with a screen where the graphical elements explicitly show the available functionalities, which encourages a more user-driven interaction. Consequently, a dialogue model designed for a vocal system may not be directly transferable to a multimodal context since the system prompts have to account for both the user's vocal input and the consequences of that input on the graphical display.

Furthermore, the cognitive load of the wizard is much higher when simulating the missing parts of a multimodal system, because he has to take into account all four elements of the interaction (the graphical interface, dialogue model, natural language interactions and interactions in other modalities) when interpreting user input, and must be fast, precise and act consistently in recurring situations.

Finally, the wizard's reaction time is important, because it should be balanced for all modalities. The problem arises when both pointing and language are represented in the system. Pointing can be processed automatically and is therefore very fast, whereas language input requires manual interpretation by the wizard, which naturally takes more time. To balance the time difference, either the speed of the pointing modality can be degraded (although this is not advisable since most users expect pointing to be fast and may react negatively to the degradation), or the wizard has to be very fast when interpreting the language input. The latter requires that the interface which the wizard uses to control the system has to allow for very efficient interpretation techniques.

4. The Extended Wizard of Oz environment

In this section, we describe the Extended WOz environment that we have used to design and evaluate a multimodal system. We describe both the hardware and software that we have used.

4.1. The hardware configuration

In our Wizard of Oz environment the evaluator of the interface, the 'user', sits at a desktop PC with a touchscreen and a wireless mouse and keyboard (Figure 2). They are also given a small lapel microphone.

The user's actions are recorded by two cameras situated on tripods. One camera faces the user and records their facial expressions as they interact with the system. The second camera is positioned to the side and slightly behind

Figure 2: View of the user's work environment.

the user in order to record which modalities they are using. Sounds and images from the computer are recorded directly as well, giving the experimenters a total of 3 views with which to work when analyzing the multimodal behavior of the user.

The wizard sits in a separate room since the user must be given the impression that he is interacting with a fully automated system. The wizard's room is equipped with 3 laptop computers (Figure 3) and an extra monitor. The first laptop (C) streams the view of the user's face and audio from the user's room (using MPEG-4 technology). The monitor (B) streams a view of the user's screen (via the VNC protocol). The second laptop (D) shows the Input Wizard's Control Interface (using VNC protocol with a specific server) which is used by the wizard responsible for processing the user's input, while the third laptop (A) shows the Output Wizard's Control Interface which is used to define and select system prompts.

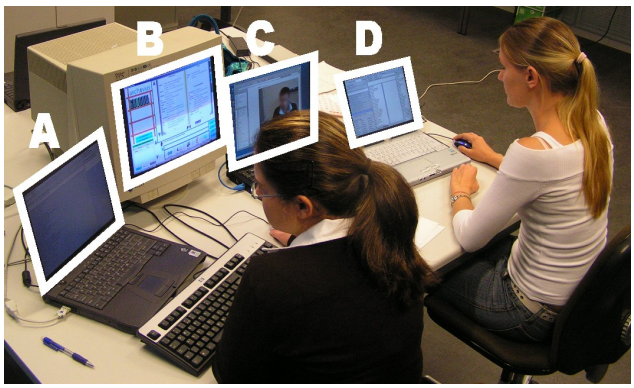

Figure 3: View of the wizard's environment. (A) Output Wizard's Control Interface, (B) Mirror of the user's desktop, (C) User's face, (D) Input Wizard's Control Interface.

4.2. The wizard's control interfaces

In order to increase efficiency and minimize the wizard's cognitive load, we propose that the wizarding tasks be shared by two wizards, each managing an important part of the interaction: the interpretation of the user's input (Input Wizard) and the control of the system's natural language output (Output Wizard).

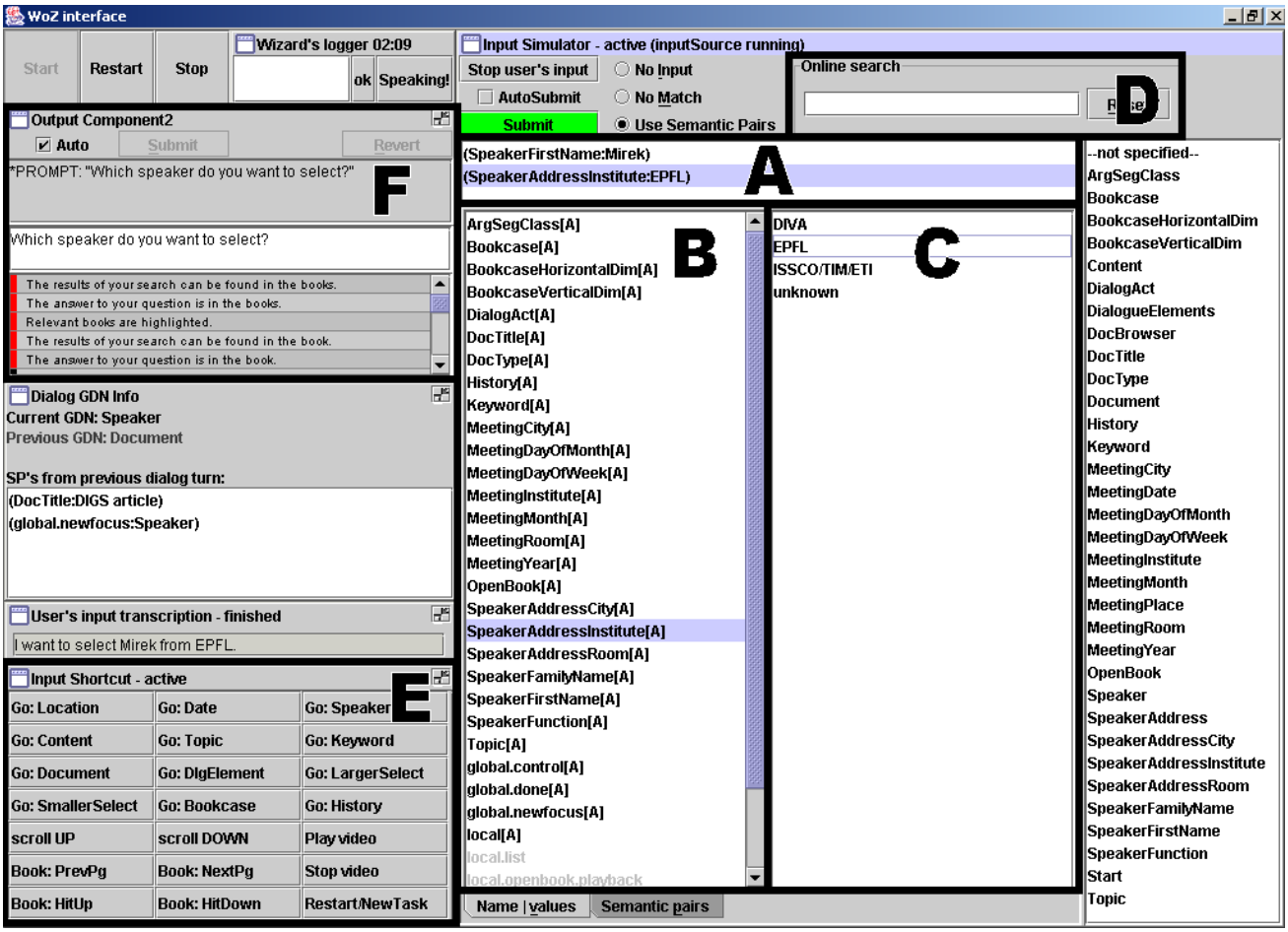

Figure 4: Input Wizard's Control Interface: (A) semantic pairs generated automatically by the system or added by the wizard; lists that allow the wizard to create a new semantic pair: (B) the list on the left shows names of semantic pairs, (C) the list on the right displays all possible values associated with the selected name of the semantic pair; (D) semantic pairs filter; (E) shortcut buttons; (F) system prompt.

The user of the multimodal system can provide his input using several modalities, making the overall interpretation of their input more challenging. Our approach to solving this problem is that each modality is processed separately, yielding (in some cases empty) sets of semantic pairs (see Section 2.2. for details about how semantic pairs represent user's input). Note that these pairs may have associated confidence scores and can be ambiguous if the input provided through a given modality is ambiguous. The semantic pairs generated by the input from each modality are then combined (and disambiguated) by the Fusion Manager into a single set. The role of the Input Wizard is to check that the set correctly represents the user's overall input and to make corrections or add new semantic pairs when necessary. In this way, the Input Wizard simulates or supervises the functionality of the SRE, NLU, and Fusion management modules in the system (or other modules processing other modalities, if they are present in the given system).

The Input Wizard's interface is shown in Figure 4. The wizard can see the semantic pairs that result from automated system processing in (A). These can be removed from the set using the '*Delete*' key. To add a semantic pair, the wizard has to select its category (name) from the list (B) first, then select the appropriate value for it (C). The list of semantic pairs is generated automatically from the task description (provided to the dialogue manager) at application start-up.

As has already been mentioned, reducing the wizard's response time is an important aspect of the system. However, the task of entering semantic pairs slows down the wizard's reaction time even more, particularly when the lists of possible interpretations contain too many entries (in our case over 2000 entries). To overcome this problem, the Input Wizard's interface contains an '*online search*' field (D) where the wizard can type in the filter that constrains the entries in the list (i.e. the name or value of the semantic pairs in the list must contain the text in the '*online search*' field). Another feature that makes the wizard more efficient is the shortcut buttons (E), which allow for entering a frequently used semantic pair in one click. The shortcuts are typically used by the wizard in situations where the user says a command (e.g. "*Show me the speakers.*"). The shortcut would produce a semantic pair such as *newFocus:Speaker* in one wizard's click. The inclusion of all of these features helped to reduce wizard response times to an average of about 2.5 seconds (see detailed response times for each modality in Figure 5).

The second wizard (the Output Wizard) controls the

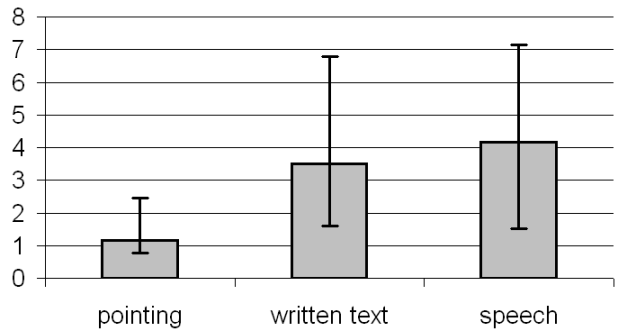

Figure 5: Input Wizard response times (in seconds) for each user input modality.

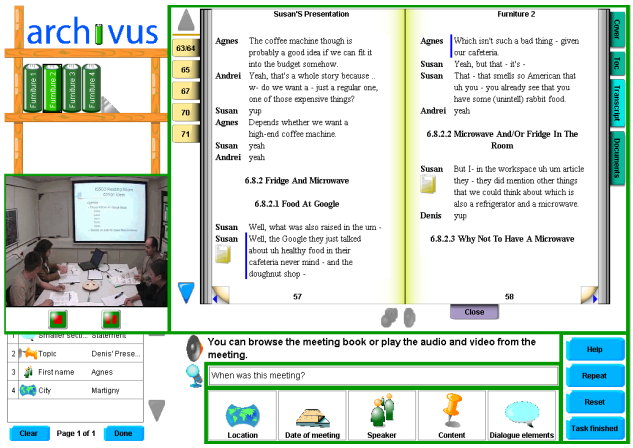

Figure 6: An example of the Archivus system interface.

system prompts using yet another interface on a different machine. This interface is a larger and more detailed version of the one shown in window (F) in Figure 4. The Output Wizard can choose to select a prompt other than that which is proposed by the system, in situations where a different prompt contributes to smoother dialogue flow. The interface provides a set of predefined prompts which can used as they are or modified 'on the fly', and new prompts can be entered freely during the dialogue. All modified and new prompts (and the corresponding dialogue system states) are logged, with the aim of automatically learning how to modify the dialogue strategy during interaction. One of our future goals is to change the prompts automatically based on the information from these log files (corpus).

5. An example: the Archivus system

We have used the methodology on a multimodal (mouse, touchscreen, keyboard and voice) system called Archivus described in detail in (Lisowska et al., 2004). This system allows users to access a multimedia database of recorded and annotated meetings. Specifically, the database contains the original video and audio from the meetings recorded in special meeting SmartRooms such as described in (Moore, 2002), electronic copies of all documents used or referred to in the meetings as well as handwritten notes made by participants during the meeting, and a text transcript of the meeting itself. In order to facilitate retrieval of information, selected annotations have also been made on the data, specifying elements such as dialogue acts, argumentative structure and references to documents, as well as the date and location of the meetings and information about the meeting participants.

Archivus was designed based on the Rapid Dialogue Prototyping Methodology (RDPM) for multimodal systems (Cenek et al., 2005) developed at the École Polytechnique Fédérale de Lausanne, and is meant to be flexibly multimodal, meaning that users can interact unimodally, choosing to use any of the available modalities exclusively, or multimodally, using any combination of the available modalities.

6. Experiments

Thus far, we have performed 3 sets of pilot Wizard of Oz experiments using the Extended WOz methodology. 40 participants were involved in the experiments, whose primary aim was to improve and fine-tune both the Archivus system and the Wizard of Oz environment.

In the final set of pilot experiments we settled on the following experimental protocol. Users were first given a consent form to sign, and a questionnaire to fill out which gathered general demographic information. They were then asked to read a set of instructions and do a short tutorial with the Archivus system using a subset of modalities. A manual for the Archivus system was also available to the user throughout the experiment. The evaluation proper was carried out in two phases, each lasting 20 minutes, where users had to answer a mixture of true-false and short-answer questions to which the answers could be found by using the Archivus system. In the first phase, the user was given the same subset of modalities that they had available to them during the tutorial. In the second phase users were given access to all of the modalities available for interaction with the Archivus system (voice, touchscreen, keyboard and mouse) and were free to choose which modalities they wanted to use. In the final part of the experiment users were asked to complete another questionnaire, whose goal was to elicit their overall opinion of the system. In cases where a novel modality such as voice was not introduced in the tutorial, the user was given a very brief second tutorial on how to use that modality before they began the second phase of the experiment. Moreover, we attempted to minimize the introduction of any bias via the experiment documents by making all of the tutorials, as well as the instructions to the user as modality-neutral or balanced as possible. The order in which the questions were given to the user was also varied to test for any influence on interaction from the order of the questions.

The data that resulted from the experiments gave useful indications of how to improve the wizard's environment, in particular highlighting which aspects can safely be fully automated to decrease overall processing time, and how to structure the information that the wizard sees and must manipulate in order to reduce the time during which they intervene. Finally, the experiments revealed several flaws in the design of the Archivus system which impeded the user's successful interaction with it. These flaws were fixed be-

tween experiment sets and the results tested in the next experiment, resulting in 3 full design-development-evaluation cycles.

The next stage of our work will involve a full-scale evaluation of multimodal interaction using the Archivus system in the Extended Wizard of Oz environment. The results of this full-scale evaluation will serve four purposes: (1) to see what modalities are used, how they are used and for what tasks, (2) to elicit natural language input in order to build robust language models, (3) to evaluate the dialogue model and (4) to evaluate the Archivus interface itself.

7. Conclusions

In this paper we have shown how the Wizard of Oz methodology can be extended for the design and implementation of multimodal interfaces. In our preliminary experiences with applying this methodology to the design of such a system (three complete design-development-evaluation cycles), we have found that a WOz setup originally designed for natural language systems has to be adapted for multimodal systems and should address (at least) the following problems:

- Defining an interface for the wizard that allows them to interpret input from different modalities and simulate system responses accurately and efficiently.
- How to augment dialogue models originally designed for voice-only interfaces for use in a multimodal context. In a graphical environment the interaction is more user- than system-driven since the user has a visual context for their interaction which makes it easier for them to decide what to do next.
- Determining the influence that each element of the system (GUI, dialogue management, natural language components and other input modalities) has on the other elements, since changing one of these elements could impact interaction with the other 3 elements as well.

Taking the time to develop a suitable and efficient interface for the wizard can be a great benefit to the quality of the overall results achieved during Wizard of Oz experiments with complex multimodal systems. It is only once the Wizard of Oz environment has been finalized that experiments can start with the real goal in mind: to elicit data that will be used to help automate the missing components of the system.

8. Acknowledgements

The work 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.

9. References

Trung H. Bui, Martin Rajman, and Miroslav Melichar. 2004. Rapid Dialogue Prototyping Methodology. In Petr Sojka, Ivan Kopeček, and Karel 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. Springer-Verlag.

Pavel Cenek, Miroslav Melichar, and Martin Rajman. 2005. A Framework for Rapid Multimodal Application Design. In Václav Matoušek, Pavel Mautner, and Tomáš Pavelka, editors, *Proceedings of the 8th International Conference on Text, Speech and Dialogue (TSD 2005)*, volume 3658 of *Lecture Notes in Computer Science*, pages 393–403, Karlovy Vary, Czech Republic, September 12-15. Springer.

Hua Cheng, Harry Bratt, Rohit Mishra, Elizabeth Shriberg, Sandra Upson, Joyce Chen, Fuliang Weng, Stanley Peters, Lawrence Cavedon, and John Niekrasz. 2004. A Wizard of Oz Framework for Collecting Spoken Human-Computer Dialogs. In *Proceedings of INTERSPEECH 2004 - ICSLP, The 8th International Conference on Spoken Language Processing*, pages 2269–2272, Jeju Island, Korea.

Nils Dahlbäck, Arne Jönsson, and Lars Ahrenberg. 1993. Wizard of Oz Studies – Why and How. In Dianne Murray Wayne D. Gray, William Hefley, editor, *International Workshop on Intelligent User Interfaces 1993*, pages 193–200. ACM Press.

Petra Geutner, Frank Steffens, and Dietrich Manstetten. 2002. Design of the VICO Spoken Dialogue System: Evaluation of User Expectations by Wizard-of-Oz Experiments. In *Proceeedings of the 3rd International Conference on Language Resources and Evaluation (LREC 2002)*.

Jan Krebber, Sebastian Möller, Rosa Pegam, Ute Jekosch, Miroslav Melichar, and Martin Rajman. 2004. Wizard-of-Oz tests for a dialog system in Smart Homes. In *Proceedings of the joint congress CFA/DAGA*, Strasbourg, France.

Agnes Lisowska, Martin Rajman, and Trung H. Bui. 2004. 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.

Kent Lyons, Christopher Skeels, and Thad Starner. 2005. Providing Support for Mobile Calendaring Conversations: A Wizard of Oz Evaluation of Dual–Purpose Speech. In *MobileHCI '05: Proceedings of the 7th International Conference on Human Computer Interaction with Mobile Devices & Services*, pages 243–246, New York, NY, USA. ACM Press.

Darren Moore. 2002. The IDIAP Smart Meeting Room. In *IDIAP-Com 02-07*, page 13.