Applying the Wizard of Oz Technique to the Study of Multimodal Systems

Daniel Salber and Joëlle Coutaz

Laboratoire de Génie Informatique, IMAG
B.P. 53 X, 38041 Grenoble Cedex, France
Phone: +33 76 51 44 40, Fax: +33 76 44 66 75
E-mail: salber@imag.fr, coutaz@imag.fr

Abstract. The Wizard of Oz (WOz) technique is an experimental evaluation mechanism. It allows the observation of a user operating an apparently fully functioning system whose missing services are supplemented by a hidden wizard. From our analysis of existing WOz systems, we observe that this technique has primarily been used to study natural language interfaces. With recent advances in interactive media, multimodal user interfaces are becoming popular but our current understanding on how to design such systems is still primitive. In the absence of generalizable theories and models, the WOz technique is an appropriate approach to the identification of sound design solutions. We show how the WOz technique can be extended to the analysis of multimodal interfaces and we formulate a set of requirements for a generic multimodal WOz platform. The Neimo system is presented as an illustration of our early experience in the development of such platforms.

Introduction

Communication between the user and the computer has been shown to be significantly enhanced when different input media are simultaneously available [16]. There is a high potential for systems allowing the use of combined input media, but our knowledge for designing, building, and evaluating such systems is still primitive. Evaluation techniques and user observation provide ways to improve the design of interactive systems. In this paper, we focus on the Wizard of Oz (WOz) experimental evaluation technique. We show how this technique can be extended to the analysis of multimodal interfaces and formulate a set of requirements for multimodal WOz platforms. The Neimo system is presented as an illustration of our early experience in the development of such platforms.

Evaluation Techniques and Wizard of Oz Experiments

Evaluation Techniques

Evaluation techniques are twofold. They may be based on predictions from theory or they may rely on experimental data.

Predictive models and techniques do not require any system implementation nor do they need effective users. Examples of such techniques include GOMS [5] and its related models such as CCT [18], theory-based models such as ICS [4], KRI [20], and the "cognitive walkthrough" method [19]. Their main benefit is to allow user interface evaluation at an early stage of the development process. However, such predictions rely on theoretical hypotheses, not on real data. Thus, predictive models and techniques may lack precision or they may be limited in scope when the case study (e.g., multimodal interaction) is not supported by the underlying theory. In addition, the setting and interpretation of such models are sometimes complex and as time consuming as an effective implementation.

At the opposite, experimental techniques deal with real data observed from real users accomplishing real tasks, and operating a physical artifact. Artifacts may be paper scenarios, mock-ups, computer system prototypes, or Wizard of Oz systems. A WOz system allows the observation of a user operating an apparently fully functioning system whose missing services are supplemented by a hidden wizard. The user is not aware of the presence of the wizard and is led to believe that the computer system is fully operational. The wizard observes the user through a dedicated computer system connected to the observed system over a network. When the user invokes a function that is not available in the observed system, the wizard simulates the effect of the function. Through the observation of users' behavior, designers can identify users' needs when accomplishing a particular set of relevant tasks and evaluate the particular interface used to accomplish the tasks.

Existing Wizard of Oz Systems and Lessons Learned

Existing WOz Systems. Most of existing WOz systems have been primarily developed to study the usage of natural languages for retrieval information systems. Telephone information services such as telephone directories, flights or trains information and reservation services, have been an interesting field for experiments [11, 23]. The experimental setup is quite simple: the wizard answers phone calls and pretends callers are talking to an automatic information system. In order to give callers the illusion that they are actually talking to a computer, the wizard's voice is filtered through a distortion system (e.g., a vocoder) that gives the voice a robotic flavour. Questions and answers are tape-recorded for later manual transcription and analysis. Other case studies involve databases or advisory systems interrogation [14, 17, 26] as well as dialogues with expert systems [8, 22]. Most of them aim at collecting vocabulary corpus in order to tune and augment the robustness of spoken or written natural language recognizers. The platform described in [7] is an attempt to support the observation of graphical direct manipulation combined to natural language. A recent paper [21] describes Turvy, an intelligent agent simulated using a WOz, that can be taught using speech and direct manipulation.

Although limited in scope, results from WOz experiments already form an interesting body of knowledge about wizards and evaluation experts.

Lessons Learned From Wizards. An interesting result from WOz observations is that wizards' tasks, although apparently simple, are cognitively expensive. The realism of the apparatus requires wizard's actions to be consistent in content, style, and pace. In particular,

• in similar contexts, a given command from the subject must trigger the same behavior from the wizard,

• response time must comply to the subject's expectation: if the wizard is too slow at reacting, the subject may avoid using simulated functions or may believe that the system is overloaded.

In summary, wizards cannot afford improvisation. To achieve an acceptable consistent behavior, wizards must be trained at well defined tasks, and must be assisted by powerful tools. To this end, some WOz systems include limited but useful mechanisms such as a set of predefined replies or menus containing pre-stored parts of answers [6].

In order to alleviate cognitive overload, recent studies suggest a two-wizard configuration where one wizard is specialized in I/O whereas the second one performs task level processing [1]. The I/O wizard acquires user's requests and transmits simulated answers; the task wizard interprets the requests translated by the I/O wizard and generates the answers to be formulated by the I/O wizard. This collaborative task sharing is more likely to guarantee consistency. It doesn't add noticeably to the response time provided that the wizards are appropriately trained. Another experiment using a two-wizard configuration has proven to be successful [10].

Tools For Analysis. Another lesson learned from WOz experiments is the need for analysis tools that support efficiently the task of the evaluation experts. Manual analysis of a large body of data requires expertise and is time-consuming. When collected data is recorded electronically, the analysis can be partly automated. For example, in natural language dialogues, the use of pronouns can be identified [6] or, as in the Wizard's Apprentice, the dialogue structure can be analysed [8]. Although the apparatus described in [15] is not a WOz platform, it provides a precise searchable record of user's behavior: events related to graphical menus and buttons are captured and aggregated into higher abstractions, then linked through timestamps to videotape frames. Although this system tracks direct manipulation actions, the salient features of user behavior must be extracted by hand.

In summary, most of existing WOz systems have been developed on a case-per-case basis and support the observation of one modality only. Similarly, automated analysis tools are limited in scope and are rarely integrated into the WOz platform from the start. Although [7] is intended for the combined use of graphics and natural language, the system is limited by technical constraints and supports one wizard only. There has been no attempt to produce a generic, reusable WOz platform that would make possible the observation and analysis of multimodal interaction. With multimodal interaction, new problems arise with specific requirements.

Requirements for a Multimodal Wizard of Oz

Requirements for multimodal WOz platforms can be defined from the users' perspective (i.e., the wizards and the evaluation experts) as well as from the software point of view. Before discussing these issues, we must introduce the notion of multimodal interaction.

Multimodal Interaction

Multimodal interaction is characterized by the possibility for the user to exploit multiple communication channels. It requires the system to represent and manipulate information at multiple levels of abstraction including the dynamic maintenance of meaning in the task domain [3, 13].

As discussed in [3, 13], multimodal interaction may be characterized along two dimensions:

• the temporal use of modalities (i.e., communication channels) which can be sequential or concurrent,

• the level of fusion between modalities: modalities may be used in a combined way (e.g., the "put that there" paradigm) or used independently.

The usage of multiple modalities along the fusion and temporal constraints have direct implications on software design as well as on users. The wizard is one such user.

Requirements From The Wizards' Perspective

From the wizard's perspective, multimodality increases the complexity of the task as well as information bandwidth.

Task Complexity. In a traditional WOz system, the wizard supplements a single missing service such as the interpretation of typed natural language. Within an interaction where the subject can use multiple modalities concurrently, the wizard must simulate more complex functions such as the synergistic combination of modalities. Thus, the wizard must have a very efficient way of observing the subject. As shown by early experiments in face-to-face communication, internal video circuits are not accurate enough to track every of the subject's actions [12]. Clearly, the wizard must also have a dedicated computer of his own. The computer is less prone to tracking failure than the human wizard and, as discussed above, it can provide ready for use answers or encapsulate elementary subject actions into higher abstractions such as commands.

Information Bandwidth. In a multimodal system, the subject has many ways for providing inputs. The same semantic content may take multiple forms. Thus, the amount of information to be conveyed to the wizard is likely to be much more important than in a traditional system. The required input processing bandwidth may become too high for a single wizard and may result in inconsistent behavior with respect to content, form, and response time. Thus a multiwizard configuration seems a reasonable way to go.

Multiwizard Configuration. The difficulty in a multiwizard configuration is to organize the collaboration between the wizards in a way that guarantees consistent behavior.

Ideally, the workload should be equally distributed among the wizards. However, the workload is difficult to estimate a priori since it heavily depends on the subject's behavior. In these conditions, it should be possible for the wizards to dynamically change their respective roles. In addition, we feel the need for a specialized wizard who would act as a supervisor. This superwizard would not accomplish any regular wizard task but would regulate wizards' behavior, monitor the session, and make proper decisions in case of system malfunction.

Task allocation among wizards is currently being investigated. Although we know that it depends on the peculiarities of the experiment, we are working on a set of general features that could help in supporting task distribution. For example, we have found useful to structure a typical wizards' task in three steps: acquisition, interpretation, and formulation.

- acquisition consists of analyzing the message issued by the subject, and of extracting the relevant information in the problem space,
- interpretation corresponds to the task level problem solving in the domain,
- formulation covers the form and emission of an answer to the subject.

This decomposition in three steps suggests the allocation of a combination of steps to different wizards. The I/O wizard and the task wizard discussed above are one such possibility. However, when multiple modalities have to be combined, the fusion process requires an extra acquisition step that complements the acquisition over each communication channel. In this case, it may be wise to define a specialized wizard for performing the fusion of multiple modalities.

Requirements From The System Perspective

The overall property of a platform for multimodal WOz experiments is performance and flexibility. Flexibility covers configuration and software communication protocol.

Performance. As discussed above, the wizard is an important element in response time conformance. However, if the system is slow, the wizard cannot compensate the flaw. Thus the system has to provide the effective foundations for efficiency.

Configuration Flexibility. Configuration flexibility covers the variable number of wizards as well as the variety of input and output devices. Multiple wizards, each equipped with a dedicated computer, are recommended when task overload must be shared. In addition, a multimodal WOz system should be flexible enough to accept new interaction devices. The adjunction of a new device should simply result in the development of a new module that would allow the subject to use the new modality and let the wizards track, interpret and simulate its operation. By doing so, devices that are not yet supported by the software under test could be conveniently simulated by a wizard.

Communication Protocol Flexibility. In an ordinary WOz system, communication between the subject's computer and the wizard's computer is a major issue. In a multimodal WOz system, this point is even more critical for two reasons. First, such a system may require multiple wizards with multiple workstations possibly exchanging information with each other. Second, a generic multimodal WOz platform requires that the information exchanged over the communication channels may be of any level of abstraction. As an illustration, consider the design of a drawing software where modalities available to the subject are mouse pointing and voice-enabled commands. Designers should be able to perform the following experiments:

- in the very first step, the tested software would implement mouse gestures only, and voice recognition would be performed by wizards. In this experiment, low level events should be transmitted to the wizards' workstation (e.g., mouse movement in terms of "x,y" coordinates along with the replication of the subject's utterances).
- in a further development stage, the software would be enhanced with a speech processing system. Events of a higher level of abstraction such as the recognized sentence and the identification of the selected object, should be sent to the wizards' computer.

This example shows the need for a flexible communication protocol, allowing communication between the subject and the wizards systems, as well as communication between the wizards. This protocol should also be flexible enough to allow easy transmission of any level of information, from low-level user events up to high-level information of any kind.

Requirements for Evaluation Experts: Powerful Analysis Tools

A long-term goal for multimodal WOz systems is to include computerized automatic evaluation tools. This objective, however, still requires theoretical research. In the short run, it is realistic to provide an integrated way to retrieve and manipulate a posteriori data collected during the experiment. With this perspective in view, we propose the following requirements:

- data should be collected on the same physical support; a support that allows collecting from different media and that offers great flexibility is a computer file. Since we intend to have a fully-computerized WOz system, integration of all history data into computer files is a natural choice.
- data should be correctly and automatically synchronized to allow real time replay and analysis.
- automatic processing should be used whenever possible; for example, if speech is recorded in history data, detection of syllables should be performed automatically while analyzing the history data.
- a powerful filtering mechanism should be available to provide an efficient way to reduce the amount of data to be analyzed, or to allow a direct focus on a particular aspect of the experiment. For example, in the case of the voice-controlled drawing software, one should be able to focus on the choice of

modalities used for drawing new objects; the analysis tool would locate the instants of the session where the user decided to draw a new object and would display the modalities used.
• versatile visualization capabilities could be very useful. For example, to study the use of an electronic pen or a mouse, evaluation experts should be able to analyze the pointer motion in terms of (x, y) coordinates as well as replaying in real time the movements of the pointer.
• detection of multiple repeating patterns of behaviour from the user is a useful evaluation strategy [24]. Since the history data is stored in a computer file, it can be processed electronically and such processing as well as statistical computations on history data should be integrated.

The Neimo Project: A Multimodal Wizard of Oz Platform

With the requirements exposed above in mind, we designed and developed Neimo, a multimodal Wizard of Oz platform, generic and extensible [2]. We have already performed early tests with Neimo. The purpose of these first experiments is to observe and analyze multimodal interaction with a mouse-based application along with speech recognition and interpretation of facial expressions [25].

System Configuration

The current configuration for our experiments with the Neimo platform consists in a workstation for the observed user, and a workstation for each wizard. All workstations are Apple Macintosh Quadras. Figure 1 shows an example configuration involving the following modalities: mouse pointing, speech, and facial expression.

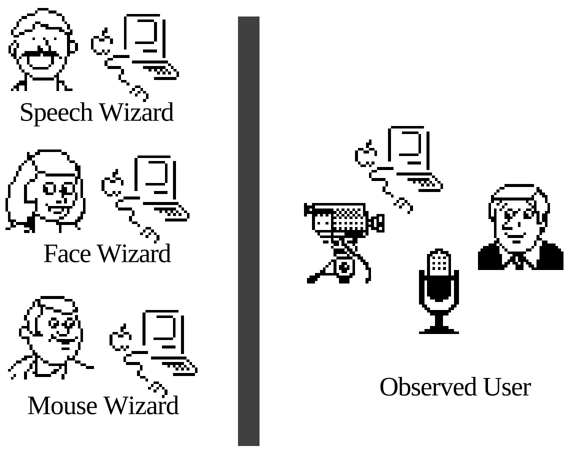

Fig. 1. An example configuration for a Neimo experiment

In this example, three wizards collaborate:

- The "mouse wizard" tracks the use of the mouse and cooperates directly with the speech wizard when fusion of a user's speech command and mouse designation is required.
- Speech recognition and interpretation is simulated by a dedicated "speech wizard". The observed user's utterances are digitized and saved in history files, using a sound digitizing board.
- A CCD camera is focused on the user's face and connected to a video acquisition board. The "face wizard" can observe the user's face digitized in real time, interpret the user's expressions, communicate the interpretations to the other wizards to help them in their task, and store the interpretations for later analysis.

Software Organization

One of our major requirements for a multimodal WOz platform is independence regarding the observed application: we want to be able to plug any client application onto the WOz platform, as long as its source code is available. To fulfill this requirement, Neimo includes a library that allows any client application to use services provided by the WOz platform. These services include:

- sending messages to the wizards, for example to request the simulation of a missing function,
- receiving messages from the wizards, that are taken into account by the client application,
- saving information in history files.

The format of the messages exchanged between the application and the wizards is defined by the client application. More precisely, a set of standard messages has been defined for usual functions required by applications and for usual modalities, but application-specific or modality-specific messages can also be defined; in this case, a specific software component must also be developed for the wizards, to interpret these messages on the wizards' side. This mechanism guarantees the flexibility of the platform, and allows for future evolution.

Therefore, Neimo accepts any application as client, and any specific modality can be integrated into the platform.

The Communication Services. The software organization of our multimodal WOz platform is built around a communication kernel (Neimo Com) as shown on figure 2. The studied application and the wizards' applications have access to the communication services via libraries.

Neimo Com is a process that can run on any workstation part of the experiment, or even on a dedicated workstation. All the messages exchanged between the observed user's computer and the wizards or among the wizards themselves are sent through Neimo Com. Thus, Neimo Com is a time base for the whole system: it time-stamps all the messages and also records pertinent information in history files. Neimo Com is in charge of dispatching the messages between the wizards: it distributes to each wizard the messages he is able to process, and collects replies to be transmitted to the observed user as well as operations the wizards want the observed application to

perform. Neimo Com also arbitrates possible conflicts, and handles network-related errors and failures.
In short, Neimo Com is the communication control center of the Neimo platform. It supports any number of wizards, and allows them to connect to the system and disconnect from it dynamically, thus allowing the wizards to come into play when needed, even during an experiment.

Neimo provides libraries to allow client access to the communication services. The library on the client application side allows sending and receiving messages, as well as saving information in history files. Since the types of messages are client-defined, they can be of any level of abstraction. On the wizards side, the library provides the following services:

- receiving messages from the observed application. A dynamic subscription mechanism allows the wizards to express interest in certain category of messages. For example, the speech wizard subscribes to the *speech* message type to receive the user's utterances. In addition, the dynamics of the subscription allows wizards to change roles in the course of an experiment.
- sending messages to the observed application. These messages may then be interpreted by the application as operations to perform; this allows remote control of the observed application.
- saving interesting information to history files.
- exchanging messages with other wizards. Up to now, only a "talk"-like exchange tool has been designed, that allows the communication of short typed messages; but we would like to provide the wizards the possibility to exchange audio, and even video messages.

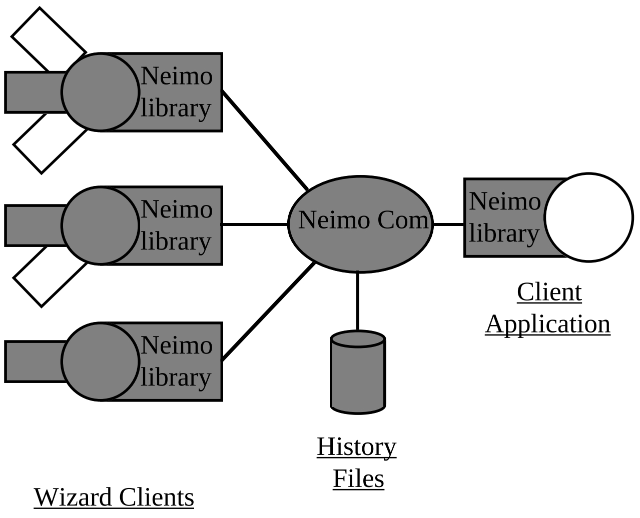

Figure 2. The software components of the Neimo platform.
Dimmed areas denote common services and white areas specific components

As a summary, the originality of the Neimo communication framework is three-fold:

- messages from different media are processed in a uniform and integrated way,

- the format of messages is not imposed by the system, but is client-defined. Thus the messages can convey information of any level of abstraction, from low-level events such as mouse clicks up to high-level commands.
- the Neimo Com communication center allows dynamic dispatching of the messages, according to their type and the subscriptions of the connected wizards.

The Wizards. Requirements we exposed above concerning the wizards are very strong. To fulfill them, our approach to the design of the wizards' software is to allow as much flexibility as possible. So, we will be able to test many configurations, and we will study a set of rules to configure most efficiently the wizards' operations.

On the wizards' side, Neimo allows for any number of wizards. In our early tests, we assign each wizard the handling of messages corresponding to a given modality. Each wizard has an application that suits his specific needs. Specific modules can be plugged onto a wizard application to provide specific services. Each different kind of message sent by the observed application has a corresponding module in one of the wizards' application. For example, the speech wizard has a speech module that handles the processing of received *speech* messages; it allows the wizard to hear the user's utterances and to perform a set of possible actions in response to a spoken command of the user. A wizard application can accept many plug-in modules; so, a wizard may be able to play many roles, at once or successively during the session. This architecture guarantees a great flexibility for wizard applications.

After the Session: Analyzing the History Data

As we emphasized above, the analysis of the session after the experiment can be a very long and heavy task. With Neimo, we provide analysis services to help this task.

All data collected during the experiment is stored in history files, and is automatically synchronized. Neimo includes a library that allows exploitation of history files. Since the messages exchanged during a Neimo session, as well as the messages saved in history files, have client-defined content (i.e. defined by the observed application and its specific wizard modules), these history files can only be exploited by client applications that are aware of this content. So, Neimo provides the "history library", which can manipulate history files, and let client applications exploit their content. It allows reading and selective browsing of history files. It also integrates a tool for detecting multiple repeating patterns of behaviour.

A Few Technicalities

The Neimo system runs on Apple Macintosh Quadra workstations connected through an Ethernet network. It is written in C and C++, and is developed with MacApp, an object-oriented application framework. The communication protocol relies on Apple Events, a powerful high-level inter-application communication protocol introduced by Apple with System 7. The history files use a QuickTime-based format; QuickTime [27] is an extension to Apple’s System 7 for manipulation and storage of time-based data from different media.

Conclusion and Perspectives

We exposed requirements for a multimodal Wizard of Oz platform, and presented Neimo, an attempt to meet these requirements. Two major areas still need further investigation. First, the organization of the wizards' work requires a lot of testing and experiments; we intend to define a set of rules that would help configure the wizards' operations. Second, the tools for the analysis of history files need further development; a long-term goal for this analysis is partial or total automation.

Acknowledgements

We would like to thank Jean Caelen of the Institut de la Communication Parlée (ICP, Grenoble, France) who provided very valuable help and advice as well as René Amalberti and Claude Valot from CERMA, Jean-Marc Francony and Evert Kuijpers from CRISS, Grenoble, who shared their skills and experiences with us.

References

1. R. Amalberti and C. Valot: "Le Magicien d'Oz", CERMA, journée du PRC Rhône-Alpes, 1993.
2. G. Ambone, B. Noz, D. Salber: "Projet Neimo, Spécifications Externes", internal technical report, équipe IHM, LGI-IMAG, 1992.
3. S. Balbo, J. Coutaz and D. Salber: "Towards Automatic Evaluation of Multimodal User Interfaces", Workshop on Intelligent User Interfaces '93.
4. P.J. Barnard, “Cognitive Resources and the Learning of Human-Computer Dialogs”, in Interfacing Thought, Cognitive Aspects of Human-Computer Interaction, J.M. Carroll Ed., MIT Press Publ., 1987, pp.112-158
5. S.K. Card, T.P. Moran and A. Newell: "The Psychology of Human-Computer Interaction", Lawrence Erlbaum Associates, 1983.
6. N. Dahlbäck and A. Jönsson: "Empirical studies of discourse representations for natural language interfaces", Fourth Conference of the European Chapter of the ACL, proceedings 291-8 1989.
7. N. Dahlbäck, A. Jönsson and L. Ahrenberg: "Wizard of Oz studies — why and how", Third Conference on Applied Natural Language Processing, Trento, Italy, 31 march—3 April 1992.
8. D. Diaper: "The Wizard's Apprentice: A Program to Help Analyse Natural Language Dialogues", People and Computers V, proceedings of the 5th Conference of the British Computer Society, 1989.
9. P. Falzon: "Ergonomie Cognitive du Dialogue", Presses Universitaires Grenobloises, 1989.
10. J.-M. Francony, E. Kuijpers and Y. Polity: "Towards a methodology for Wizard of Oz experiments", Third Conference on Applied Natural Language Processing, Trento, Italy, 31 march—3 April 1992.
11. N. Fraser, N. Gilbert and C. McDermid: "The Value of Simulation Data", Third Conference on Applied Natural Language Processing, Trento, Italy, 31 march—3 April 1992.

12. R.S. Fish, R.E. Kraut, R.W. Root: "Evaluating Video as a Technology for Informal Communication", in the CHI'92 Conference Proceedings, ACM Press Publ., 1992, pp. 37-48
13. A. Gourdol, L. Nigay and D. Salber: "Multimodal Systems: Aspects of Event Fusion and a Taxonomy", to be published in IFIP Conference '92 Proceedings.
14. R. Guindon and K. Schuldberg: "Grammatical and Ungrammatical Structures in User-Adviser Dialogues; Evidence For Sufficiency of Restricted Languages in Natural Language Interfaces to Advisory Systems", proceedings of the 25th Annual Meeting of the Association for Computational Linguistics.
15. M.L. Hammontree, J.J. Hendrickson, B.W. Hensley, "Integrated Data Capture and Analysis Tools for Research and Testing on Graphical User Interfaces", in the CHI'92 Conference Proceedings, ACM Press Publ., 1992, pp. 431-432
16. A.G. Hauptmann: "Speech and Gesture for Graphic Image Manipulation", CHI '89 Proceedings, pp. 241-245.
17. A. Jönsson and N. Dahlbäck: "Talking to A Computer is Not Like Talking To Your Best Friend", Proceedings of the Scandinavian Conference on Artificial Intelligence '88, pp. 53-68.
18. D. Kieras, P.G. Polson: "An Approach to the Formal Analysis of User Complexity", International Journal of Man-Machine Studies, 22, 1985, pp. 365-394
19. C. Lewis, P. Polson, C. Wharton, J. Rieman: "Testing a Walkthrough Methodology for Theory-Based Design of Walk-Up-and-Use Interfaces", CHI '90 Proceedings, pp. 235-241.
20. J. Löwgren and T.Nordqvist: "A Knowledge-Based Tool for User Interface Evaluation and its Integration in a UIMS", Human-Computer Interaction—INTERACT '90, pp. 395-400.
21. D. Maulsby, S. Greenberg, R. Mander: "Prototyping an Intelligent Agent through Wizard of Oz", InterCHI'93 Proceedings, pp. 277-284.
22. Y. Polity, J.-M. Francony, R. Palermiti, P. Falzon, S. Kazma: "Recueil de dialogues homme-machine en langue naturelle écrite", Les Cahiers du CRISS, n° 17, 1990.
23. M. Richards & K. Underwood: "How Should People and Computers Speak to Each Other", Proceedings of Interact '84 — First IFIP Conference on Human-Computer Interaction, pp. 268-273.
24. A. Sochi & D. Hix, "A study of Computer-Supported User interface Evaluation Using Maximal Repeating Pattern Analysis", in Proceedings of the CHI'91 Conference, ACM Press, pp. 301-305, 1991.
25. M. Turk and A. Pentland: "Eigenfaces for recognition", Journal of Cognitive Neuroscience, Vol. 3, No. 1, pp. 71-86, 1991.
26. S. Whittaker and P. Stenton: "User Studies and the Design of Natural Language Systems", Fourth Conference of the European Chapter of the ACL, proceedings 291-8 1989.
27. QuickTime Reference Guide, QuickTime CD, Apple Computer Inc.