WOEB: Rapid Setting of Wizard of Oz Experiments and Reuse for Deployed Applications

Andrea Bellucci, Paolo Bottoni, and Stefano Levialdi
Dipartimento di Informatica
Università Sapienza di Roma, Italy
{bellucci,bottoni,levialdi}@uniroma1.it

ABSTRACT

[1]We describe an approach and an environment for setting up Wizard of Oz experiments to test multimodal interaction (through mobile devices) for multimedia content delivery. This is based on a metamodel and a reference client-server architecture, whose implementation is discussed. The approach allows progressive refinement of the Wizard of Oz environment into a deployed application, through the use of a single metamodel, to be progressively specialized, and a service-oriented approach which allows the substitution of implemented services to simulated ones. A visual interface is provided for the definition of the wizard interface, of the metadata relating content delivery and possible interactions on it, and of some aspects of the client interaction.

1. INTRODUCTION

The increasing capabilities of processing power and memory and the high resolution displays of current mobile devices, together with the expansion of available bandwidth and powerful compression techniques, enable their users to enjoy advanced multimedia experiences. On the other hand, PDAs, Tablet PCs, and even smart-phones, are embodying different types of sensors and interaction strategies, including pen-based interaction, VoIP support, GPS, etc. These also allow new methods of interaction, combined in a multimodal way, to steer the usage of multimedia material.
Interaction with these devices differs from the traditional desktop experience with multimedia content, as it occurs in less protected environments affected by noise, unusual postures, disruptions in availability of certain channels, e.g. WiFi or GPS coverage [6]. Multimodality is becoming a common interaction paradigm due to its naturalness. However, the peculiarities of multimodal interactive systems make it difficult to gather information from the use of modalities that can be employed for improving the user interfaces [1].
It is therefore important to be able to test the ways in which people can interact with new mobile applications before their final deployment. Moreover, users' needs can be detected using evaluation techniques [7] dealing with real data, gathered from the observation of users accomplishing real tasks, while operating on a physical mobile device [14]. System prototyping contributes to demonstrating functionality and appearance of a software application, including usability testing [8, 10, 13].
Wizard of Oz (WOz) techniques [11, 14] are a popular way to perform this testing. Similarly to the character in F. Baum's story, in a WOz experiment a human *wizard* simulates the behaviour of a partially implemented multimodal system. The subjects of the experiment must believe they are interacting with a real, fully implemented one, thus maintaining a natural interaction behaviour. The wizard, unknown to the subject, monitors the user through a dedicated computer program, connected to the observed system over a network. When the subject invokes a function that is not implemented by the observed system, the wizard simply simulates its effect. In this way, designers can capture specific interaction phenomena, without the mediation and distortions of human-human communication. However, setting up a WOz experiment may be hard in that the working environment for the wizard has to provide complete control on the user input and to present a general view of the available actions and of their relevance to the current interaction state, in order to activate them coherently. This requires an organization of content and interface which takes into account the device characteristics, so that the relevant content has to be adapted to the device physical context as well as to the user task context. On the other hand, this same information will have to be used in the final application, so that it would be desirable to be able to employ the same logic for the construction of the WOz interface and of the final application.

While many WOz experiments are performed using *ad hoc* machinery, we propose WOEB (Wizard of Oz Experiment Builder) as a general framework for the construction of the WOz logic and interface in terms of services. The deployment of the final application can proceed through the progressive refinement of the available services, and the substitution of the WOz services with those running the actual application. WOEB is based on a fragment of a metamodel for multimodal adaptive multimedia information systems and relies on services for the identification of the relevant content on the server-side, to be transmitted to the client-side on the mobile device, in the format which is most suitable to the user preferences and the device capabilities. At this stage of design, our approach focuses only on a subset of the possible interaction modalities typically related to interaction with hand-held devices; for instance we do not take into account bimanual interaction. We have tested WOEB in a scenario where a technician must be remotely instructed on troubleshooting hardware. We have run two experiments:

[1]Partially funded by MIUR: Projects CHAT and PRIN 2006.

in the first one, the system provides the technician with some predefined graphic or textual content about the hardware, and interaction is performed by selecting special areas (hotspots or hyperlinks) in the content, along with oral commands. In the second, in conjunction with speech inputs, 2D physical labels are placed directly on hardware components [9]. Using the built-in camera of the mobile device, the user can detect the label of a desired component and request further information from the system.

2. RELATED WORK

WOz experiments can be run on both high- and low-fidelity prototypes. In [10], the authors present the WOZ PRO system, a pen-based software environment enabling both the creation of low-fidelity prototypes, and the conduction of WOz experiments. Since they do not aim at the iterative refinement of the prototype into a fully functional multimodal application, they do not exploit an architectural metamodel.

The development of a complete multimodal system starting from WOz experiments is described in [3], where the experiment is conducted in a different environment, in order to study integration of speech and gesture pointing in the construction of collaborative story-telling.

Momento [6] supports *on-site* evaluation of ubiquitous computing applications, addressing issues identified in interviews with developers of mobile technology and observations of daily studies [5, 4]. Like WOEB, Momento consists of a set of configurable tools that can be used without writing source code. Momento employs the text messaging and media messaging services of mobile devices to share content between the mobile application and the wizard's platform. Conversely, WOEB exploits a wider range of communication channels (such as Wi-Fi or GPRS connections) to share multimedia content, including audio or video streams.

The Ozlab [13] tool investigates different forms of user participation in the early phases of systems development and supports the WOz protocol to perform experimental studies. Ozlab was conceived to handle both natural language interfaces as well as GUI interactivity testing. It allows text output typed by the wizard or previously defined or voice output, wizard-made or pre-recorded.

Bunt *et al.* [2] distinguish between content *adaptation* and *presentation*, where the first involves deciding what content is most relevant and how to structure this content in a coherent way, while the second involves deciding how to adapt the presentation of the selected content to the user. WOEB supports both processes, leaving different degrees of freedom to designers and wizards to force specific forms of adaptation, while leaving others to the implemented components.

The applications of the suggested environment are that of adaptive hypermedia, for which several general models are being proposed. In [12], a set of models for the adaptation process are identified, namely: *domain*, *navigation*, *user*, *integration*, *presentation*, *adaptation* and *context*, and three types of adaptable elements: *content*, *navigation* and *presentation*. Our metamodel has several aspects in common with this, but offers reuse, allowing binding of model components to specific implementation elements.

3. A METAMODEL FOR MULTIMEDIA INFORMATION SYSTEMS

In WOEB, the construction of WOz modules logic and interface relies on a metamodel and a reference architecture. A typical WOz environment provides a set of tools organized in a Client-Server architecture. The Server side is the Wizard module, hosting a human operator who simulates the not yet implemented functionalities of the system. The Client side is the multimodal human-computer interface deployed on the mobile device.

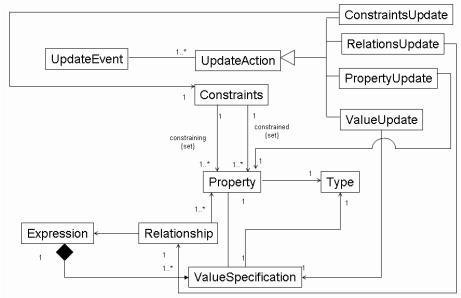

Figure 1: The basic metamodel for the definition of models.

All packages of the WOEB metamodel import a basic `Model` package, a fragment of the UML 2.1.1 metamodel presented in Figure 1, which allows the definition of (simple or derived) typed properties, paired with expressions for their evaluation. A model can contain different relationships between properties, and the presence of some properties can be constrained by others. Finally, the model must provide ways to update its composition, upon receiving some triggering event.

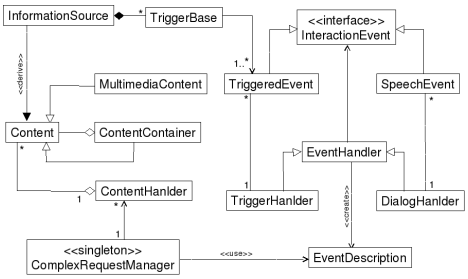

Figure 2: A fragment of the complete metamodel for the definition of interactive applications.

Figure 2 illustrates the relationship between information in the client model and in other packages, in particular the multimedia content model and the adaptation model, by defining suitable classes as containers of properties. Hence, the client will host *information sources* derived from some *content* on the server, organized into a composite structure. The content is served by a *content handler*, according to an adaptation process performed by a *request manager* which interprets event descriptions received by specific *handlers* for client-generated events. The content delivery to the client creates an *information source*, which provides support to further interactions in the form of some *base*, triggering *interaction events* which are composed and interpreted as requests for new content.

4. SYSTEM COMPONENTS OUTLINE

The WOEB environment is a set of tools for the rapid setup of WOz experiments. We identified here three main actors: the WOz experiment designer, the wizard and the subject of the experiments.

The *WOz Builder Module* offers a GUI for specification of XML-based descriptions of the WOz Server and Client components. An *XML Interpreter* processes the XML description to produce the Server and Client widgets and functionalities, according to the metamodel. The *WOz Server Module* represents the system used by the human wizard to lead the experiments. It allows the wizard to serve users requests with a simple and intuitive interface and integrates the data collection facilities. The server module also implements the server-side communication mechanisms for sending and receiving data from the *WOz Client Module*, which is the multimodal application running on a mobile device used by the subject of the experiment.

5. DESIGNING THE PROTOTYPE OF A MULTIMODAL SYSTEM

In the current implementation, the *WOz Builder* enables the definition of the information sources, which represent the way the Client displays content. As the interaction model accommodates the combination of speech input with triggering events, the *WOz Builder* has to present an interface for the definition of such events.

In the first experiment, WOEB is used for developing a multimodal mobile application where interaction is based on the selection (through a pointing device) of special areas in the information source, an image in our case. We do not describe the second experiment, due to lack of space. The designer loads the source (the image itself) in a dedicated panel of the *WOz Builder* and defines the *TriggerBase* as sensitive areas. In the current implementation, an area can be a hyperlink, if the information source is a hypertext, or a polygon for a graphical information source.

By selecting an interactive area, the designer associates a description of the relative content, which will be sent to the client following a user request. This involves the specification of attribute-value pairs describing the content's behavior relative to the user's delivery context. Such attributes can be used to guide the adaptation of content to be delivered to the mobile device (via a DISelect engine[2] which represents the realization of the *ComplexRequestManager* in our implementation of the metamodel). For example *content_manifestation* admits as values *audio*, *text* or *graphics*, while *request_context* defines the type of user request a particular content is designed to serve, e.g. *how* or *what*. Different request context types can be defined at the time of WOz modules definition. The builder can automatically organize the layout of the WOz Server GUI, so that the buttons are distributed over columns labeled by the request context.

In Figure 3, the information source is an image of a computer mother-board on which a technician has to intervene supported by the mobile system. The CPU socket area has been defined as a hot spot. On the client-side, a user can interact with the system selecting it and asking "what is this?" or "how can I use this one?". The designer has thus defined two request context types, called "what" and "how"(Figure 3), and assigned two "what" or "how" label to relative content. This information can then be used by the DISelect engine to filter a subset of content, matching this attribute. The designer can also define general purpose content, not associated with any interactive area, in order to manage oral user interactions or unpredicted events. Once all the elements are specified within the *WOz Builder*, definitions (in terms of interface artifacts and services) are stored in an XML archive. This platform-independent description of the logic and interface of the WOz modules can be processed by customized engines to generate the effective WOz Server and Client components. In the current implementation the *XMLInterpreter* co-generates both the WOz Server and WOz Client interfaces and services.

[2]http://www.w3.org/TR/cselection/.

5.1 WOz Server Module

This module was automatically generated by the *XMLInterpreter*, starting from the definition contained in the XML archive previously mentioned. Information sources are placed in a tabbed pane including an appropriate content viewer. For each information source a dedicated panel is created for content selection. Each panel contains buttons for each trigger base element in the information source (the motherboard image in Figure 4) that can be used by the human wizard to call the associated content handler (e.g. the *DISelect Engine*).

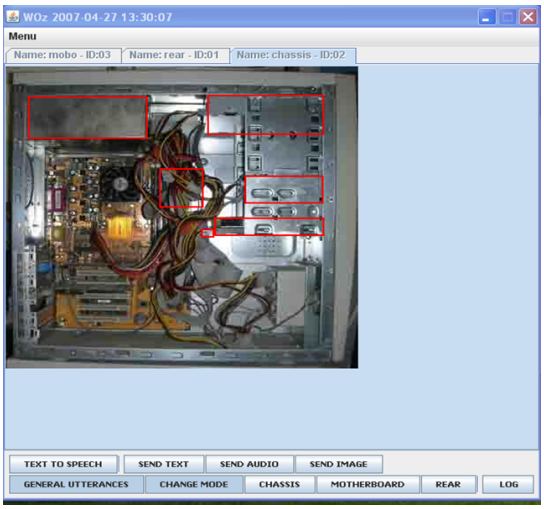

Figure 4: A portion of the WOz server interface.

In order to simplify content selection by the human wizard, buttons are organized in columns labeled with the different request contexts. Contexts are used by the DISelect engine to preliminarily filter the content to be retrieved, depending on the mobile device. This feature can be disabled during WOz module definition: for example if some components are fully implemented. The *WOz Server* is designed to be a starting point to build a real adaptive multimodal system. A data communication interface, handling data exchange between client and server, is implemented as well as a *Timestamp Agent* tracing the temporal order of the events. To deploy the final system, only services that are currently embodied by the human wizard have to be implemented: for example an Adaptive Dialog Manager, or a module to

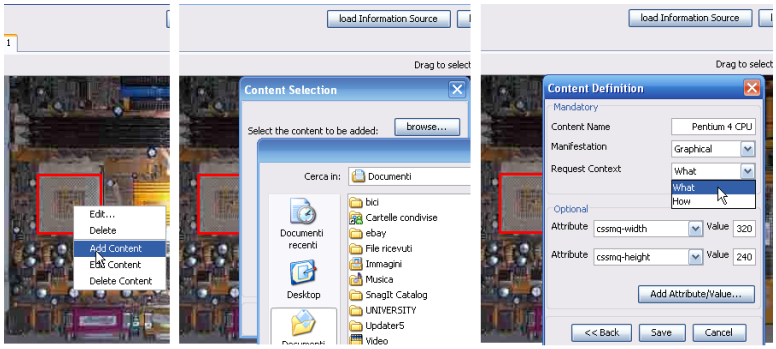

Figure 3: *WOz Builder*: adding a content

manage the multimodal fusion of pointing and speech events through the *Timestamp Agent*. One can also replace existing modules with customized ones: as an example the *DISelect Engine* can be substituted by another implementation of the *Content Handler* interface, simply by redefining the process of attribute definition and the content description within the *WOz Builder*.

5.2 WOz Client Module

In a way similar to the Server Module, the WOz Client Module is automatically generated by the *XMLInterpreter*. This module is essentially a data viewer, which can display content sent by the server and manage user pointing and oral interaction. Pointing interaction is captured through the trigger base, while audio inputs are processed by a *Speech Capture* engine. A description of a triggered event is sent, while the captured speech is streamed directly to the Dialog Manager on the server (e.g. the human wizard).

6. CONCLUSIONS

We presented WOEB (Wizard of Oz Experiments Builder), a framework and environment for setting up of Wizard of Oz experiments by defining modules interface and services. WOEB encourages the development of multimodal systems for managing multimedia information through the progressive refinement and implementation of components and services. The adoption of a common metamodel for the definition of the Wizard of Oz experiments and the deployment of the final application favors the rapid identification of problem sources and the reuse of components and services for different application scenarios. The first tests with WOEB show its efficacy in rapid setting of experiments for exploring different interaction modes. Further initial experiments have been performed, only partially reported here due to lack of space.

7. REFERENCES

[1] R. Bernhaupt, D. Navarre, P. Palanque, and M.A. Winckler. Model-based evaluation: A new way to support usability evaluation of multimodal interactive applications. In E. Law, E. Hvannberg, G. Cockton, and J. Vanderdonckt, editors, *Maturing Usability: Quality in Software, Interaction and Quality*, volume HCIS, pages 96–122. Springer, 2008.

[2] A. Bunt, G. Carenini, and C. Conati. Adaptive content presentation for the web. In *The Adaptive Web*, volume 4321 of *LNCS*, pages 409–432. Springer, 2007.

[3] S. Carbini, L. Delphin-Poulat, L. Perron, and J.E. Viallet. From a Wizard of Oz experiment to a real time speech and gesture multimodal interface. *Signal Processing*, 86(12):3559–3577, 2006.

[4] S. Carter, S. R. Klemmer, and J. Mankoff. Exiting the cleanroom: On ecological validity and ubiquitous computing. *HCI*, 23(1):47–99, 2008.

[5] S. Carter and J. Mankoff. When participants do the capturing: the role of media in diary studies. In *(CHI 2005)*, pages 899–908. ACM, ACM Press, 2005.

[6] S. Carter, J. Mankoff, and J. Heer. Momento: support for situated ubicomp experimentation. In *CHI 2007*, pages 125–134. ACM, 2007.

[7] Scott Carter and Jennifer Mankoff. Prototypes in the wild: Lessons from three ubicomp systems. *IEEE Pervasive Computing*, 4(4):51–57, 2005.

[8] B. Hartmann and S. R. Klemmer. Reflective physical prototyping through integrated design, test, and analysis. In *UIST06*, pages 299–308. ACM, 2006.

[9] L. E. Holmquist. Tagging the world. *interactions*, 13(4):51–ff, 2006.

[10] C. Hundhausen, S. Trent, A. Balkar, and M. Nuur. The design and experimental evaluation of a tool to support the construction and wizard-of-oz testing of low fidelity prototypes. In *IEEE Symposium on VL/HCC 2008*, pages 86–90, 2008.

[11] J. F. Kelley. An iterative design methodology for user-friendly natural language office information applications. *ACM Trans. Inf. Syst.*, 2(1):26–41, 1984.

[12] D. Schwabe P. Seefelder de Assis. A semantic meta-model for adaptive hypermedia systems. In *AH 2004*, volume 3137 of *LNCS*, pages 360–365. Springer, 2004.

[13] J.S. Pettersson. Ozlab: a system overview with an account of two years of experiences. In *HumanIT 2003*, pages 159–185, 2003.

[14] D. Salber and J. Coutaz. Applying the Wizard of Oz technique to the study of multimodal systems. In *EWHCI*, volume 753 of *LNCS*, pages 219–230. Springer, 1993.