

Generation of Device Independent User Interfaces

Oscar Mayora-Ibarra
ITESM, Campus Cuernavaca
omayora@campus.mor.itesm.mx

I Introduction

It is inevitable that in the near future, mobile phones, PDA's, remote controls, wearable computers and other new portable devices will continue to keep our hands and pockets busy. Internet access, home-device controlling, learning and teaching material, interpersonal communications, entertainment and business applications will be ubiquitously accessible and not farther than a mouse / button "click" or voice-command distance. The promise of information anytime, anywhere will become an everyday reality. Specialized and easy to use information appliances, such as intelligent hand-held devices, will proliferate and practically dominate the future of human-machine interactions. In this new scenario dominated by pervasive and ubiquitous computing technologies, the efficient operation of these new hi-tech portable devices and the rapid assimilation of their usage will highly depend on the simplicity of their user interfaces (UIs). For this reason, human-centred design will be at the basis of future UI developments. Constructing intuitive interfaces will be a major requirement independently of the technological expertise level of users. Some key-design aspects for achieving complete user satisfaction will include concepts like interface personalisation, augmented reality, multimodal interaction, context adaptation and non-obtrusive computing (invisible computers).

II Device Independent User Interface Generation

According to the different moments of man-machine interaction, there will be the necessity of alternation or combination of different interaction modalities. In some cases, speech interaction may be preferred to manual input (i.e. when eyes and hands are busy) or combination of gesture and voice modalities may be used as an alternative to remote controls in home-in applications (i.e. reinforcing a command by simultaneous speech and pointing interactions). Besides, control and controllable devices should communicate among each others in order to perform automatic actualisations and continuous update of their status. The availability of multiple control devices will lead to the construction of families of UIs that may run under different platforms. In this way, it will be possible to perform same tasks by using different appliances that may vary in physical characteristics (like size, screen type, available input modality, etc). For example, in some cases the interface will be physically visible and will require the attention of the user (like a display in a mobile device) and in other cases the interaction will be realised through not visible sensors positioned either inside a room environment or within context sensitive every-day-life artefacts.

The former envision suggests a major restructuring in the way we design interfaces. Instead of the traditional one device – one UI concept, control of devices will be done via other devices, which may even be remote from the controlled device. Not only will be the UI front-end thus separated from the back-end, also the interaction techniques may vary with the properties of the control device. These properties include, for

instance, the available interaction modalities, display dimensions, and the available software platform. Designing a user interface supporting this kind of foreign control mechanism sets new requirements for the format used to specify the UI. The uncertainty about the final features of the device rendering the UI calls for a very generic, flexible and device independent format. The task becomes even more complicated when inherently different modalities need to be supported, like graphical and speech-driven UIs. Traditional approaches would suggest to write different code versions for each possible type of interaction modality. The former solution will require to write as many versions as there were modalities, which will originate great maintenance and coherence problems. These considerations motivate the development of a more generic methodology that allows for the generation of user interfaces regardless of their physical characteristics or the available interaction modality.

A generic interface is one whose aspect may vary in different devices while its functionality prevails in any of them. The design considerations involved in such kind of approach may permit to define interaction features that may be rendered according to the target device characteristics. Devices with larger screens may display the information and allow interaction in a completely different way than in smaller interfaces or screen-free devices (like the telephone). In any case, the functionality of the system may allow to perform the same task regardless the device that is used to do it. For this purpose, a set of interaction elements with different functionality must be identified and associated to different rendering formats.

The description of a generic UI implies to follow a design philosophy that overlays the different interaction features presented in different platforms. In this sense, several solutions have been proposed that assail the problem in different levels. The most of the them include XML-oriented approaches to create generic interfaces, most notably XUL used in the XPToolkit project of Mozilla, the XML application [1] used with MOCA [2] and the UIML language promoted by Harmonia [3]. Other solutions have been found using Java-based approaches with the shortcoming of limited adaptability to target-specific rendering formats. In our case, the intention was to found the best language option to create a generic vocabulary that would be adequately rendered to visible graphic interfaces as well as to invisible voice-based interfaces. We found UIML among the mentioned languages to be the one that closest maps to our design expectations.

One of the main features of UIML is that it allows to separate the content of an interface from its presentation format. Every document written in UIML is divided into five main sections: *structure*, *content*, *behaviour*, *style*, and *peers*. The first four sections are grouped into the interface section and are used to specify different parts of a user interface, including an object (part) tree (*structure*), the text, graphics, audio etc. (*content*), the events, conditions and actions needed when interacting with the UI (*behaviour*), and the properties of the widgets and a mapping to a more specific widget (*style*). The last section, *peers* section, describes how the parts and events are mapped to widgets and constructs in the target UI platform. There may be peers subsections (presentation and logic) for each specific target platform, and for efficiency these may be contained in separate files. An example of the same interface content with different presentation formats is shown in figure 1. In this example, the content of a common

source written in UIML is adapted and downscaled to different devices according to their physical constraints. In this way, a HTML-based interface may include all the graphical information (like pictures and detailed presentation of items) a WML version may limit the interface to only present the more essential information and a VoiceXML presentation may generate a conversation dialog for interacting. Designing for both speech and graphical interfaces is a major topic within the use of a generic UI. A proposed solution to that matter may be found in [4].

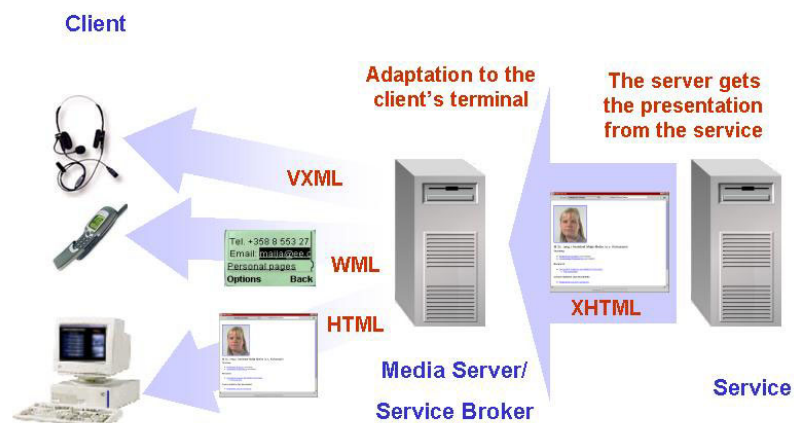


Figure 1: Presentation of a family of UIs.

III Future Work

Future work will be done in order to improve the generic vocabulary proposed in [5]. Besides, comparison of different transcoding techniques may be still carried out in order to define the advantages and disadvantages of every one of them. New target formats (like WML) should be still implemented in our tool and a demo environment should be completed. Profiling and personalisation options should be added to the user interfaces and finally multimodal support may be included as an interaction possibility.

References:

- [1] Román M., Beck J., and Gefflaut, A., "A device-independent representation for services", *Proceedings of the Third IEEE Workshop on Mobile Computing Systems and Applications* (pp. 73-82), Monterey, CA: IEEE
- [2] Beck J., Gefflaut A., and Islam N., "Moca: A service framework for mobile computing devices.", *Proceedings of the ACM International Workshop on Data Engineering for Wireless & Moblie Access (MobiDE)*, Seattle, WA: ACM Press (pp. 62-68).
- [3] Abrams M. and Phanouriou C., "UIML: An XML language for building device-independent user interfaces.", *Proceedings of XML 99*, Philadelphia 1999
- [4] Plomp, Johan; Mayora-Ibarra, Oscar; Yli-Nikkola, Heli , "Graphical and speech-driven user interface generation from a single source format "Proc. of AVIOS 2001. San Jose, US, 2 - 4 Apr 2001. San Jose, US (2001).
- [5] Plomp J., Mayora-Ibarra O., "A generic widget vocabulary for the generation of graphical and speech-driven user interfaces." *International Journal of Speech Technology*, V(5), Issue 1, pp.39-47, Kluwer Academic Publishers, January 2002.