MULTIMOMODALITY AND MULTI-PLATFORM INTERACTIVE SYSTEMS

Fabio Paternò
ISTI-CNR, Via G.Moruzzi 1, 56124 Pisa, Italy

Abstract: This paper discusses how to take into account multimodality when designing multi-platform interactive systems. In particular, it focuses on graphical and vocal interaction and discusses how to obtain interfaces in either one modality or their combination starting with logical descriptions of the tasks to perform. It also introduces how such an approach can enable migratory interfaces exploiting various modalities.

Key words: Multimodality, Multi-platform User Interfaces, Model-based Design, Migratory Interfaces

1. INTRODUCTION

One of the current main challenges for designers and developers of interactive systems is how to address applications that can be accessed through a variety of devices that can vary in terms of interaction resources (screen size, processing power, modalities supported, …). In order to address such challenges model-based approaches have raised a good deal of interest: the basic idea is to have concepts and languages that express aspects relevant to user interaction in a device independent manner and then to support a design process that is able to identify usable solutions taking into account the features of the devices available.

In this context many modalities can be considered. In the paper we focus on the graphical and vocal modalities and discuss their impact in the design process following a model-based approach aiming at developing multi-modal interfaces in Web environments. The approach discussed is supported

by a tool, TERESA (Mori, Paternò, and Santoro, 2003) providing various transformations among different abstraction levels. The abstraction levels considered are: the task level, where the logical activities are described including the objects necessary to their accomplishment; the abstract interface level, a conceptual description of the user interface in a modality independent language; the concrete level, where the concrete design decisions are described; and lastly the user interface. The design process goes through these levels, and information regarding the target platforms and devices is considered at each level even if the approach allows designers avoid dealing with low-level implementation details. By platform we mean a class of systems that share the same characteristics in terms of interaction resources. They range from small devices, such as interactive watches to very large flat displays. Examples of platforms are the graphical desktop, PDAs, mobile phones and vocal systems. The modalities available have an important role in the definition of each platform.

Multimodality is important in this approach because of the intrinsic differences among modalities. For example, the vocal channel is more suitable for simple or short messages, for signaling events, immediate actions, to avoid visual overloading and when users are on the move, whereas the visual channel is more useful for complex or long messages, for identifying spatial relations, when multiple actions have to be performed, in noisy environments or with users stationary.

This paper discusses how multimodality can be considered when designing multi-device interfaces through an approach that takes into account the potential interactive devices since the early phases (for example, identifying the tasks suitable for each platform). The approach allows designers to avoid a plethora of low-level details related to all the potential devices by using conceptual descriptions that are automatically translated into the user interface implementation.

2. MAPPING TASKS INTO MULTIMODAL INTERFACES

Our approach exploits a number of transformations that allow designers to move through various views of interactive systems. TERESA allows designers to focus on the logical tasks to accomplish and then transform their descriptions into a user interface description, which incorporates the design decisions but is still in a format modality independent. This is then used to derive a modality-dependent interface description that is used to generate the final code of the user interface. For each logical level considered a XML-based language has been defined. While the languages for the task and the

abstract interface descriptions are the same for all the platforms, each platform has its own description language for the concrete interface. The advantage of this approach is that designers can focus on logical aspects and make design decisions without having to deal with many low-level implementation details. The environment provides support to make concrete design decisions that take into account the target platforms and generate the code where all the low-level details are specified. This type of support is particularly useful when a variety of devices should support access to the interactive application. On the other hand, it is worth noting that in order to make effective decisions designers should be aware of the target platforms and modalities even since the early stages of the design process. They do not need to know the implementation details of all the targeted devices but they still should know their main features (through the platform concept).

Indeed, there are tasks that may be meaningful only when using some specific platform or modality. For example, watching a long movie makes sense only in a multimedia desktop system whereas accessing information from a car in order to know directions to avoid a traffic jam can be done only through a mobile device and if this task should be performed while driving it can be supported only through a vocal interface. The description of an object can be easily communicated through the graphical channel with the support of images whereas it can require long and less effective texts through a vocal device. It is said that “a picture is worth a thousand words” but people do not like to hear so many words. Identifying spatial relations such as how to get in a certain position can be immediately communicated through a graphical interface whereas may be difficult to explain precisely through a vocal interface.

The available modality may also have an impact on how to accomplish a task, for example a vocal interface or a graphical mobile phone interface can require to perform sequentially tasks that can be performed concurrently in a desktop graphical interface.

In TERESA a user interface is structured into a number of presentations. A presentation identifies a set of interaction techniques that are enabled at a given time. The presentation is structured into interactors (logical descriptions of interaction techniques) and composition operators that indicate how to put together such interactors. While at the abstract level such interactors and their compositions are identified in terms of their semantics in a modality independent manner, at the concrete level their description and the values of the attributes depend on the available modality.

TERESA is able to generate code in Web user interface languages for different modalities: XHTML, XHTML Mobile Profile, VoiceXML, X+V (a combination of XHTML and VoiceXML that supports multimodal

interaction). In the case of both vocal and graphical support the multimodality can be exploited in different manners. The modalities can be used alternatively to perform the same interaction. They can be used synergistically within one basic interaction (for example, providing input vocally and showing the result of the interaction graphically) or within a complex interaction (for example, filling in a form partially vocally and partially graphically). Some level of redundancy can be supported as well, for example when feedback of a vocal interaction is provided both graphically and vocally.

The composition operators indicate how to put together such interactors and are associated with a communication goal. A communication goal is a type of effect that designers aim to achieve when they want to structure presentations. Grouping is an example of composition operator that aims to highlight that a group of interface elements are logically related to each other. It can be implemented in the graphical channel through one or multiple attributes (fieldset, colour, location ..) whereas in the vocal channel it is possible to group elements by inserting a sound or a pause at the beginning and the end of the grouped elements. In case of multimodal interfaces we have to consider the actual resources available for the modalities. It is different to design a multimodal interface for a desktop and one for a PDA because the graphical resources available are different. Thus, the grouping on a multimodal desktop interface should be mainly graphical and the use of the vocal channel can be limited to provide additional feedback of the selected elements whereas the grouping of a multimodal PDA interface should be mainly vocal and using the graphical channel only for important information or for supporting or explicative information.

If we consider an interactor the reasoning is similar. For example, the single selection interactor is implemented differently depending the modality and the cardinality of the potential choices. A graphical desktop has a lot of space available and so it is able to support a long list (for example a list of countries at world level), eventually providing some support to scroll it but in the case of small devices this is not possible. Thus, it is possible to consider splitting the single selection into two or more selections. The first selection is among groups of elements and the second one allows the section of the desired element within a smaller group. This can be implemented through either a graphical or a vocal mobile device. If multimodality can be supported for this type of interaction then in the case of the desktop system it can be used to provide some explanations about the purpose of the selection in order to avoid screen cluttering. In the case of the PDA it can be used to support the two resulting interaction and the associated feedback in either modality.

3. MIGRATORY MULTI-MODAL INTERFACES

Another interesting result of the model-based approach discussed is the possibility of supporting migrating interfaces: interfaces that can be transferred from one device to another one during a user session without having to restart from scratch. This type of interfaces makes sense only if the various interfaces are able to adapt to the hosting devices. This can be achieved through the approach discussed, which allows designers to obtain different versions of application interfaces tailored for each platform. Such versions should be managed by a migration server. The goal of the migration server is to maintain information of the devices available for the migration service and their characteristics, receive requests of migration, identify the target device of the migration and activate a new instance of the interface for the identified device. Such interface should already have the state of the results of the user interactions with the source device.

This implies a number of operations:

- The state of the user interactions should be stored;
- The server should be able to convert such state into the state for the target device interface;
- The server should be able to identify the specific presentation that should be activated in the target device in order to allow continuity of interaction;

These types of operations can be performed because of the logical information that is created during the model-based design process. The tasks supported by each interface device can vary or can substantially change how they are supported. The migration server knows the associations between tasks and interactors and between interactors and interface elements for each platform. Thus, for each user interaction it can identify the corresponding task and then identifies how it is supported by the target device and associate to the corresponding interface elements the state generated by the results of the user interactions in the source device. In addition, the knowledge of the last task performed on the source device is information useful also for identifying the presentation in the target device that should be activated as a result of the migration because the user expects to carry on its session from that point.

This type of approach can be exploited also in multimodal applications. It enables multimodal migrating services that, for example, allow the user to

start the registration to the service through a desktop system and, in case he realizes that it is getting late, he can carry on the registration on the move using a PDA until he reaches his car where he can use a vocal interface to complete it. All this activity can be performed without having to restart the registration procedure from the beginning at each device change.

4. CONCLUSIONS

One of the main current issues for designers of interactive systems is how to address the wide variety of potential interaction platforms that can be used to access their applications. Multimodality plays an important role in characterizing such platforms.

The paper provides a description of an approach that aims to provide a solution to such issues providing support for designing multi-device, multi-modal interfaces. Such an approach exploits the semantic information associated with the potential user interfaces at run time to support dynamic migration even through various modalities. A number of tools supporting such an approach are under development in my group at ISTI-CNR. TERESA is publicly available at http://giove.isti.cnr.it/teresa.html

ACKNOWLEDGEMENTS

This work has been supported by the EU IST Project CAMELEON (http://giove.isti.cnr.it/cameleon.html) and the EU Network of Excellence SIMILAR (http://www.similar.cc). I also thank Silvia Berti for her work on supporting vocal interfaces.

REFERENCES

R.Bandelloni, S.Berti, F.Paternò, Mixed Initiative, Trans-Modal Interface Migration, Proceedings Mobile HCI 2004, Springer Verlag LNCS.

S. Berti, F. Paternò, Model-based Design of Speech Interfaces, Proceedings DSV-IS 2003, Springer Verlag, LNCS 2844, pp.231-244.

G. Mori, F. Paternò, C. Santoro, Tool Support for Designing Nomadic Applications, Proceedings ACM IUI'03, Miami, pp.141-148, ACM Press.