TERESA: An Environment for Designing Multi-Device Interactive Services

Silvia Berti, Giulio, Mori, Fabio Paternò, Carmen Santoro

ISTI-CNR, Via G. Moruzzi, 1
56100 Pisa, Italy
{Silvia.Berti, Fabio.Paterno, Carmen.Santoro}@isti.cnr.it
http://giove.isti.cnr.it

Abstract. In this paper we illustrate TERESA, an authoring environment for the design of multi-device interfaces, which supports specifications of logical descriptions of an interactive system in device-independent languages . The tool contains an intelligent engine supporting a number of transformations in order to move across logical descriptions and, consequently, generate the user interface for various types of platforms.

1 Introduction

With the advent of the wireless Internet and the rapidly expanding market of smart devices, designing interactive applications supporting multiple platforms has become a difficult issue. One fundamental issue is how to support software designers and developers in supporting *nomadic users*, who can access their applications through multiple devices from different locations.

Model-based approaches [7] could represent a feasible solution for addressing such issues: the basic idea is to identify useful abstractions highlighting the main aspects that should be considered when designing effective interactive applications. Indeed, the approach proposed in this paper extends previous work in the model-based design area in order to support development of multi-device user interfaces through logical descriptions and associated transformations.

TERESA (Transformation Environment for inteRactivE Systems representAtions) supports transformations in a top-down manner, providing the possibility of obtaining interfaces for different types of devices from logical descriptions. It differs from other approaches such as UIML [1], which mainly consider low-level models. XIML [9] has similar goals but there is no publicly available tool supporting it. Some usability criteria have been incorporated into the tool transformations from task to user interface. This means that the tool is able to provide suggestions for selecting the most appropriate interaction techniques and ways to compose them. Such transformations guarantee a consistent design because the same design criteria are applied in similar situations.

2 Logical Design of Multi-Device Interfaces

Logical design of multi-device interfaces can be supported by a number of steps that allow designers to start with an overall envisioned task model of a nomadic application and then derive concrete and effective user interfaces for multiple devices:

– *High-level task modelling of a multi-context application*. In this phase designers develop a single model for the possible contexts of use and roles involved. Such models are specified using the ConcurTaskTrees (CTT) notation [7], which also allows indicating the platforms suitable to support each task.

– *Developing the system task model for the different platforms considered*. Here the task model is filtered according to the target platform and, if necessary, further refined thus obtaining the system task model for the platform considered. We mean for platform a set of devices that share similar interaction resources (for example, the desktop, the PDA, the vocal). Such filtering can be done automatically or manually or in a mixed manner.

– *From system task model to abstract user interface*. Here the goal is to obtain an abstract description, modality-independent, of the user interface composed of a set of presentations structured by means of interactors composed through various operators.

– *User interface generation*. In this phase we have the generation of the user interface. This phase is completely platform-dependent and has to consider the specific properties of the target device.

In the following section we provide the reader with more details about how such transformations have been taken into account and implemented within the tool.

3 TERESA

A number of main requirements have driven the design and development of TERESA. First of all, we aimed at a tool able to support different levels of automation ranging from completely automatic solutions to highly interactive solutions: the mixed initiative paradigm was then implemented in the tool. Secondly, it is model-based, as we judged more feasible handle the issues raised by multi-device user interfaces by logical views of the activities to support. In addition, each abstraction level considered can be described through an XML-based language, in order to enhance interoperability with other environments. Moreover, we aimed at providing designers with a tool leveraging *flexible development*, as our approach aims to be comprehensive and to support various possibilities, including both when different sets of tasks can be performed on different platforms (then, the task model will be the starting point) and when only a set of devices belonging to the same platform are considered (then, the logical interface description will be the starting point). Finally, we first targeted *Web-based* applications because they are the most common, although the approach can be easily extended to other environments (Java, etc.).

In TERESA some transformations are available, allowing to move from a task model specification to the generation of the user interface. More specifically, from the analysis of the CTT nomadic task model it is possible to derive an interactor-based description of the associated Abstract User Interface (AUI), composed of a number of *presentations* and *connections*. Each presentation defines a set of interaction techniques perceivable by the user at a given time, while the connections define the dynamic behaviour of the UI, by indicating what interactions trigger a change of presentation and what the next presentation is. Each presentation is defined in terms of *interactors* classified depending on their semantics and possibly composed through some *composition operators*. Such operators have been defined depending on the communication effect that they support: *grouping*, indicating a set of interface elements logically connected to each other; *relation*, highlighting a one-to-many relation among some elements; *ordering*, when some kind of ordering among a set of elements can be highlighted; *hierarchy*, different levels of importance can be defined among a set of elements. The specification is in turn the input for another transformation which yields the related concrete user interface for the specific media and interaction platform selected (a number of parameters for customising the concrete user interface are made available to the designer in this phase). Lastly, the tool automatically generates the final UI for the target platform. In the following sections we further detail such a process.

3.1 From Task Models to User Interface Implementation

The first step of the process is translating the CTT task model into an equivalent representation identifying sets of interactions that should be supported by each abstract user interface presentation.

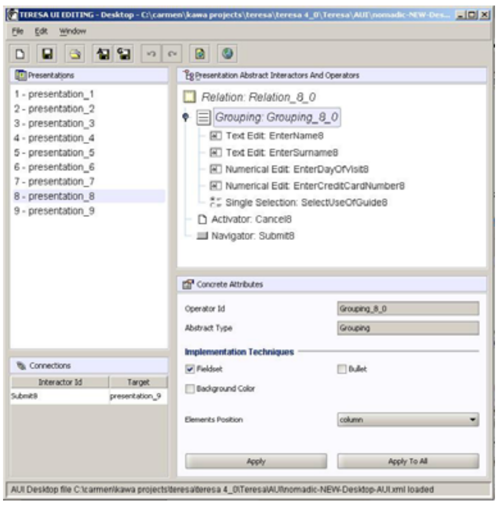

Fig. 1. Panel for setting properties of composition operators and interactors.

Both the conditions for moving across various presentations and the composition operators used in the abstract user interface are derived from the CTT task model.

The next step is translating it into a concrete user interface, in which a specific technique is used for implementing each component of the presentations for the currently selected platform. In Figure 1 you can see the window through which the designer can customise the presentations that are going to be generated by the tool. For example, the designer can decide to implement the grouping operator by means of a fieldset (see Fig.1) in the case of a desktop platform. In addition, the possible techniques differ depending on the platforms. For instance, the global parameters that are available to designers for customising the user interface might vary according to the platform (in the desktop system parameters such as the background picture, colour of the text, etc. are available, whereas in vocal devices they can be used to define welcome messages, use of barge-in options, synthesis and recognition properties, etc.).

Fig. 2. The same application with different user interfaces depending on the interaction platform.

In Fig. 2 we show examples of user interfaces derived by applying the described method to a museum application. The presentations refer to a user accessing information contained in the Marble Museum of Carrara town. As you can note, there are some differences concerning the presentations on the different platforms: for instance, for implementing the grouping operator on the desktop platform a list of graphical buttons is used, whereas on the cellphone a list of bullets is visualised, and in the vocal interface there are sounds that delimit the grouped elements.

4 Conclusions and Future Work

In this paper we have illustrated the main features of the TERESA tool, an environment supporting the design and development of multi-platform user interfaces through a number of transformations that can be performed either automatically or through interactions with the designer. The tool can be freely downloaded at http://giove.isti.cnr.it/teresa.html.

While the current version supports the design and development of graphical and vocal interfaces for various platforms (in XHTML, XHTML Mobile Profile, VoiceXML), we are working for developing new versions able to support generation of user interfaces using a broader set of modalities and their combinations (using SVG, X+V, …).

References

1. Abrams, M., Phanouriou, C., Batongbacal, A., Williams, S., Shuster, J. UIML: An Appliance-Independent XML User Interface Language, Proceedings of the 8th WWW conference, 1999.
2. Bandelloni, R., S. Berti, F. Paternò “Mixed-Initiative, Trans-Modal Interface Migration”, Proceedings Mobile HCI 2004, Glasgow, September 2004, Lecture Notes Computer Science 3160, pp.216-227, Sprinter Verlag.
3. F.Correani, G.Mori, F.Paternò, Supporting Flexible Development of Multi-Device Interfaces, Proceedings EHCI-DSVIS’04, Hamburg, July 2004, Springer Verlag, Lecture Notes Computer Science.
4. G. Mori, F. Paternò, C. Santoro, “CTTE: Support for Developing and Analysing Task Models for Interactive System Design”, IEEE Transactions on Software Engineering, pp.797-813, August 2002, Vol. 28, No. 8, IEEE Press.
5. Mori, G. Paternò, F. Santoro, C. Design and Development of Multi-Device User Interfaces through Multiple Logical Descriptions, IEEE Transactions on Software Engineering, Vol.30, N.8, August 2004, pp.507-520, IEEE Press.
6. Mullet, K., Sano, D. Designing Visual Interfaces. Prentice Hall, 1995.
7. Paternò, F. Model-Based Design and Evaluation of Interactive Application. Springer Verlag, ISBN 1-85233-155-0, 1999.
8. Paternò, F., Leonardi, A. A Semantics-based Approach to the Design and Implementation of Interaction Objects, Computer Graphics Forum, Blackwell Publisher, Vol.13, N.3, pp.195-204, 1994.
9. Puerta, A., Eisenstein, J. XIML: A Common Representation for Interaction Data, Proceedings ACM IUI 2001, pp.214-215.