End-User Customization of Multi-Device Ubiquitous User Interfaces

Fabio Paternò, Giuseppe Zichitella
HIIS Laboratory – CNR-ISTI
Via Moruzzi 1, 56124 Pisa, Italy
{fabio.paterno, giuseppe.zichitella}@isti.cnr.it

ABSTRACT
In this paper we discuss an approach to supporting end-users in customizing multi-device ubiquitous user interfaces. In particular, we show a tool allowing end-users to customize desktop-to-mobile adaptation by exploiting model-based descriptions in the MARIA language. Some results are presented along with indications for future work.

Author Keywords
End-user Development, Ubiquitous Applications, Multi-Device Environments, Adaptation.

ACM Classification Keywords
H.5.2 User Interfaces.

INTRODUCTION

One of the main issues in current technological settings is how to design and develop interactive applications that can be accessed through a wide variety of devices (ranging from small watches to very large screens, including various types of smartphones, PDAs and Digital TVs). This is particularly important in Web application, which are the most common applications.

One important research area in this context is the model-based approach, in which declarative descriptions of the user interface are used in order to avoid dealing with a plethora of low-level implementation details associated with the wide number of available devices and implementation languages. Despite such potential benefits, its adoption has mainly been limited to professional designers, but new solutions are recently emerging that are able to extend such approaches in order to achieve natural development by enabling end-users to develop or modify interactive applications still using conceptual models, but with continuous support that facilitates their development, analysis, and use [1].

Pre-proceedings of the 5th International Workshop on Model Driven Development of Advanced User Interfaces (MDDAUI 2010): Bridging between User Experience and UI Engineering, organized at the 28th ACM Conference on Human Factors in Computing Systems (CHI 2010), Atlanta, Georgia, USA, April 10, 2010.

End-User Development [3] (EUD) can be defined as a set of methods, techniques, and tools that allow users of software systems, who are acting as non-professional software developers, at some point to create, modify or extend a software artefact. End-users have already difficulties with single device applications, thus it is easy to understand how such difficulties increase when considering applications for multi-device environments. This is one further reason for providing better support for EUD in ubiquitous applications.

The vision of ubiquitous computing [9] is that the users operate in intelligent environments, which are aware of users' needs and able to assist, even proactively, the users in performing their activities and reaching their goals. To this end, one important aspect is the possibility for a user surrounded by multiple devices to freely move about and continue the interaction with the available applications through a variety of interactive devices. Indeed, in such environments one big potential source of frustration is that people have to start their session over again from the beginning at each interaction device change. Continuous task performance implies that interactive applications be able to follow users and adapt to the changing context of use while preserving their state. Thus, migratory user interfaces require integrated solutions able to address state persistence and user interface adaptation when the user changes the device.

Model-based languages are utilized at design time to help the user interface designer cope with the increasing complexity of today's applications and contexts. The underlying user interface models are mostly used to generate a final user interface code, which is then executed at run time. Nevertheless, approaches utilizing the models at run time are receiving increasing attention. We agree with Sottet et al. [8], who call for keeping the models alive at run time to make the design rationale available and show a solution for this purpose.

In the following we present some research work that exploits model-based approaches for multi-device ubiquitous applications. We show how we have enriched a software model-based platform for migratory user interfaces with a new tool for desktop-to-mobile adaptation, called parametric bidimensional semantic redesign. One of its features is that it allows the end-users to customize the adaptation process. We present some initial results and then discuss how we plan to extend them.

MIGRATORY USER INTERFACES

Migration is the result of two main features: state persistence across multiple devices and adaptation to the device interaction resources. They have to be supported while users interact with the applications made available by the intelligent environment. For this purpose, we have designed and developed a migration architecture [5], which supports a number of reverse and forward transformations that are able to transform existing desktop Web applications for various interaction platforms and support task continuity. The basic assumption is that there exists a huge amount of easily accessible content for desktop Web applications, which can be processed and transformed to support migratory interfaces, even across non-Web implementation languages. The advantage of this solution with respect to others (e.g. [4]) is that it does not require that the applications be implemented using a particular toolkit in order to make them able to migrate.

In this environment the client devices subscribe to the migration service by running a migration client that provides the environment with information regarding the device characteristics. The devices access Web applications through the migration server, which includes proxy functionalities. Migration can be triggered either by the user or it can be automatically triggered by the smart environment when some specific event (such as very low battery or connectivity) is detected, or in a mixed solution in which the environment suggests possible migrations and the user decides whether or not to accept them.

When the user accesses the application through an interaction platform other than the desktop, the server transforms its user interface by building the corresponding logical description and using it as a starting point for creating the implementation adapted to the accessing device. Figure 1 shows how adaptation is obtained. There are three main phases: reverse engineering, semantics-based adaptation, and generation. In the first phase, the tool automatically builds the logical description of the accessed page. It has rules able to handle HTML and CSS tags and associate them with the corresponding logical elements. For example, if DIV, or FIELDSET or IFRAME tags are used then it recognises that there is a group of logically connected elements in the page. We call the adaptation module semantic redesign since its purpose is to change the design still considering the interaction semantics of the implementation elements that are specified in the corresponding logical description. In addition to interface adaptation, the environment supports task continuity. To this aim, when a request for migration to another device is triggered, the environment also takes the state of the user interface, which depends on the user input (elements selected, data entered, …) and identifies the last element accessed in the source device. Thus, when a logical version of the interface for the target device is generated, it also contains the state detected in the source device version so that the user inputs (selections performed, data entered, …) are not lost. In the last phase, the user interface implementation for the target device is generated and activated remotely at the point corresponding to the last basic task performed in the initial device.

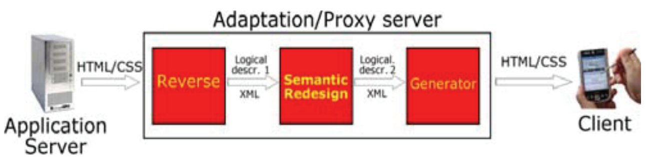


Figure 1. The main phases of the adaptation process.

In the process of creating an interface version suitable for a platform different from the desktop, we use a semantic redesign module. This part of the migration environment automatically transforms the logical description of the desktop version into the logical description for the new platform. Therefore, the goal of this transformation is to provide a description of the user interface suitable for the new platform. This means that intelligent rules are used for adapting the description of the user interface to the new platform taking into account its capabilities (e.g.: using interface elements that are more suitable for the new platform) but ensuring at the same time that the support for the original set of tasks is maintained. This solution allows the environment to exploit the semantic information contained in the logical description. In this case the semantic information is related to the basic tasks that the user interface elements are expected to support.

This software architecture for migratory user interfaces currently uses MARIA [7], a recent model-based language, which allows designers to specify abstract and concrete user interface languages according to the CAMELEON Reference framework [2]. This language represents a step forward in this area because it provides abstractions also for describing modern Web 2.0 dynamic user interfaces and Web service accesses. In its first version it provides an abstract language independent of the interaction modalities and concrete languages for graphical desktop and mobile platforms. In general, concrete languages are dependent on the typical interaction resources of the target platform but independent of the implementation languages.

In MARIA an abstract user interface is composed of one or multiple presentations, a data model, and a set of external functions. Each presentation contains a number of user interface elements (interactors) and interactor compositions (indicating how to group or relate a set of interactors), a dialogue model describing the dynamic behaviour of such elements, and connections indicating when a change of presentation should occur. The interactors are classified in abstract terms: edit, selection, only_output, control, interactive description, etc.. Each interactor can be associated with a number of event handlers, which can change properties of other interactors or activate external functions.

END-USER ADAPTATION CUSTOMIZATION

In the research on migratory user interfaces, one issue that we are considering is how to provide users with more control on the migration process in order to improve its usability. In this context more control can mean various things. One important aspect is control on the rules that drive adaptation to the various platforms (the most common case is desktop-to-mobile adaptation). For example, the adaptation engine is able to split the desktop pages when they require considerable amount of interaction resources but some users may like to have more control on the splitting algorithm.

In particular, we have designed a new tool for adaptation: *Parametric Bidimensional Semantic Redesign*. It supports adaptation from desktop-to-mobile devices and overcomes limitations of previous approaches in the area [6] because it allows users to configure the adaptation process and provides more control on costs calculation and the adaptation results. For example, while previous solutions calculated the screen space requested by the user interface elements mainly in terms of its vertical use, the new algorithm calculates both the horizontal and the vertical consumption of screen space.

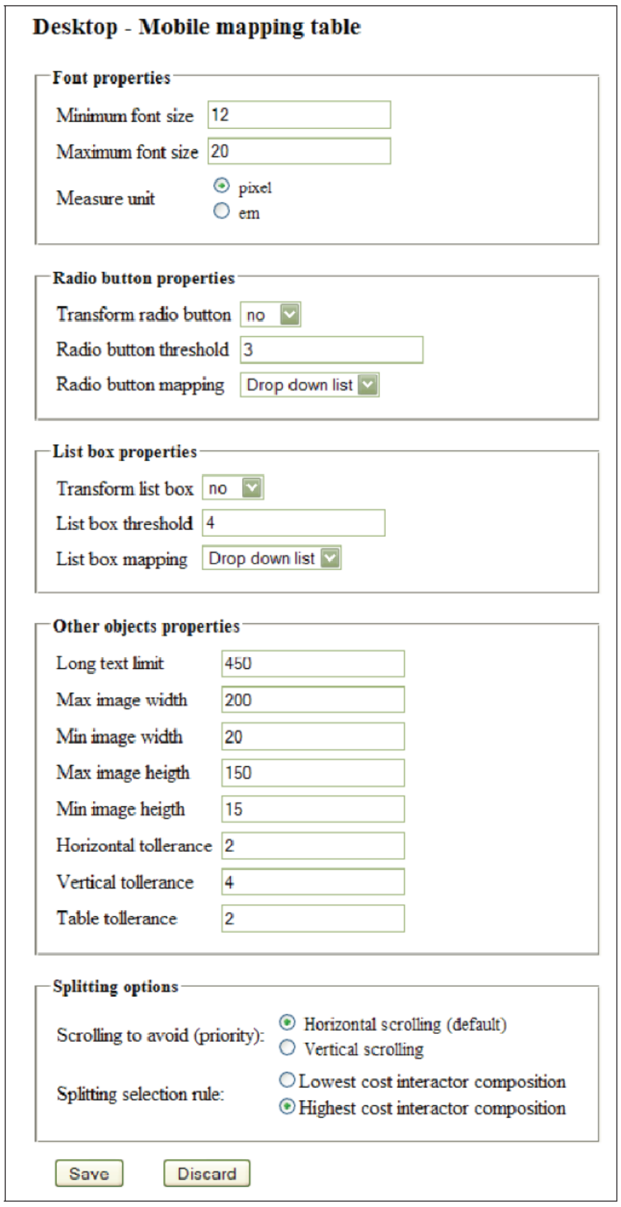
The adaptation tool takes as input the concrete description of a desktop user interface in the MARIA language and goes through a number of steps. For each step a number of specific rules are applied. First, it performs some basic transformations: if the user provides preferences regarding the minimum and maximum fonts for the target device then it transforms all the textual content in order to fit in the given range. Next, it calculates the cost of all the interactors and composition operators in the provided specification. If the resulting total cost is sustainable for the target device then the corresponding logical description is generated otherwise it starts the process to reduce the cost in order to make it sustainable. First, basic elements are adapted for the target device: the images are reduced in space while preserving their aspect ratio, some interactors are replaced with others that are semantically equivalent but needs less screen space, long texts are reduced in such a way that the part exceeding a limit is shown only on request, image and text in tables are reduced in size. After these basic transformations the overall cost is calculated again and if it is not yet sustainable by the target device then the part related to page splitting is activated. The purpose of this phase is to split the original desktop presentation into two or more presentations, which are sustainable for the target mobile device. For this purpose the algorithm considers the interactor compositions, and associates some of them to newly generated mobile presentations, removing them from the current presentation in order to decrease its overall cost.

The elements that determine the cost of the interactors are: the font attributes (size, style, type), the vertical and horizontal space required by a text, image dimensions, interline value, interactor type, and so on

Figure 2 shows the user interface that allows end users to configure the adaptation process. The various parameters are grouped according to the related user interface aspect considered. For the fonts, it is possible to specify the minimum and maximum font in the target device, and the associated measure unit. For the radio buttons it is possible to indicate whether they should be transformed into an interactor that supports the same semantics but with using less space screen. In this case, it is possible to specify the threshold, in terms of number of choice options, which should trigger the transformation and the type of interactor to use for its replacement. Similar parameters are available for the list boxes. Other parameters concern the maximum number of characters for a text, maximum and minimum dimensions for images. These parameters determine the cost of rendering a presentation. This cost is compared with the overall sustainable cost in the target device, which is given by the screen resolution multiplied by horizontal and vertical tolerance. The higher the tolerance coefficient values are, the more scrollable the generated user interface will be. This means that end users have the possibility to specify to what extent the adapted content will be scrollable in the target device. The table tolerance provides an additional factor to consider when calculating the sustainable cost. In practise, this means that when there are tables, more scrolling will be acceptable before deciding to split the presentation.

The customization interface also allows the user to indicate two additional parameters: what type of scrolling (horizontal or vertical) to avoid has the priority, and the splitting algorithm version to apply. Indeed, the tool supports two ways to determine how splitting should be performed. In both cases it analyses the cost of the composition operators (grouping or relation), which includes those of the composed interactors, and the cost of the tables (both data and layout tables). Then, the decision of the set of elements to allocate to the newly generated mobile presentation is given in one case by the most expensive element. In the other case the algorithm first calculates what elements are able to make the current presentation sustainable by the target device if removed, and then selects among them the one that has the lowest cost. The rationale for this second option is that it allows users to obtain a sustainable presentation by removing the least amount of information possible, thus preserving as much as possible the original design.

In terms of results of the adaptation process we have conducted a study comparing our tool with two publicly available tools for desktop-to-mobile adaptation: Mowser (http://mowser.com) and Skweezer (http://www.skweezer.com). The results were encouraging because our tool has shown to be more flexible since it allows end users to customize the adaptation parameters and is able to adapt a higher number of types of interface elements than the other two tools (e.g. tables and long texts do not receive specific adaptation transformations with the other two tools).


Desktop - Mobile mapping table

Font properties

Minimum font size	12
Maximum font size	20
Measure unit	[✓ pixel] em

Radio button properties

Transform radio button	no
Radio button threshold	3
Radio button mapping	Drop down list

List box properties

Transform list box	no
List box threshold	4
List box mapping	Drop down list

Other objects properties

Long text limit	450
Max image width	200
Min image width	20
Max image heigth	150
Min image heigth	15
Horizontal tollerance	2
Vertical tollerance	4
Table tollerance	2

Splitting options

Scrolling to avoid (priority):	[✓ Horizontal scrolling (default)] Vertical scrolling
Splitting selection rule:	Lowest cost interactor composition [✓ Highest cost interactor composition]

Figure 2. The customization user interface.

CONCLUSIONS

Ubiquitous environments call for adaptive systems in order to adapt to the varying interaction resources. Model-based approaches can provide useful support in this context. However, there is a need for providing users with more control on ubiquitous interfaces, according to the end-user development paradigm.

In this paper we have presented first results that allow end-users to customize desktop-to-mobile adaptation in order to change the results that can be obtained by automatic user interface generation.

We plan to further extend this work in various directions. The customization user interface can be improved in order to make the effects of the various customization parameters more understandable. In addition, in this work we have considered only desktop-to-mobile adaptation but other types of transformations can benefit from the approach proposed, e.g. graphical-to-vocal adaptation.

ACKNOWLEDGMENTS

This work has been supported by the EU project OPEN (http://www.ict-open.eu/)

REFERENCES

1. Berti, S., Paternò, F., Santoro C., “Natural Development of Ubiquitous Interfaces”, Communications of the ACM, September 2004, pp.63-64, ACM Press.
2. Calvary, G., Coutaz, J., Bouillon, L., Florins, M., Limbourg, Q., Marucci, L., Paternò, F., Santoro, C., Souchon, N., Thevenin, D., and Vanderdonckt, J. 2002. The CAMELEON reference framework. CAMELEON Project. Deliverable 1.1
3. Lieberman, H., Paternò, F., Wulf W. (eds), End-User Development, Springer Verlag, ISBN-10 1-4020-4220-5, 2006.
4. Melchior, J., Grolaux, D.,Vanderdonckt, J.,Van Roy, P., A Toolkit for Peer-to-Peer Distributed User Interfaces: Concepts, Implementation, and Applications, pp. 69.78, EICS’09, July 15–17, 2009, Pittsburgh, Pennsylvania, USA.
5. Paternò, F., Santoro, C., Scorcia, A., Ambient Intelligence for Supporting Task Continuity across Multiple Devices and Implementation Languages, the Computer Journal, the British Computer Society, 2009.
6. Paternò, F., Santoro, C., Scorcia A Automatically Adapting Web Sites for Mobile Access through Logical Descriptions and Dynamic Analysis of Interaction Resources. AVI 2008, Naples, May 2008, ACM Press, pp. 260-267.
7. Paternò F., Santoro C., Spano L.D., "MARIA: A Universal Language for Service-Oriented Applications in Ubiquitous Environment", ACM Transactions on Computer-Human Interaction, Vol.16, N.4, November 2009, pp.19:1-19:30.
8. Sottet J., Ganneau V., Calvary G., Coutaz J., Demeure A., Favre J., Demumieux R.: Model-Driven Adaptation for Plastic User Interfaces. INTERACT (1) 2007: 397-410.
9. Weiser M., "The Computer for the 21st Century" - Scientific American Special Issue on Communications, Computers, and Networks, September, 1991.