
UIDLs for Ubiquitous Environments

Fabio Paternò & Carmen Santoro

ISTI-CNR.

Via Moruzzi 1.

Pisa, 56124 Italy

{fabio.paterno, carmen.santoro}@isti.cnr.it

Abstract

In this paper we discuss some issues that seem particularly relevant for the next generation of UIDLs for ubiquitous environments. In particular, starting with an analysis of the state of art in the area of XML languages for multi-device environments we indicate areas in which further research work is necessary in order to obtain more general solutions able to address the complexity underlying ubiquitous environments.

Keywords

Ubiquitous environments, Usability and Accessibility, Guidelines, End user development,

ACM Classification Keywords

H5.m. Information interfaces and presentation

Introduction

Recent years have seen the introduction of many types of interactive devices (e.g. cellphones, PDA's, WebTV, etc.) and the availability of such a wide range of devices has become a fundamental challenge for designers of interactive software systems. Users wish to be able to seamlessly access information and services regardless of the device they are using, even when the system or the environment changes dynamically.

To address such issues a number of XML-based languages (such as TERESA XML, USIXML, XIML) have been proposed to represent user interfaces at different abstraction levels (through model-based approaches) and then enable transformations to generate corresponding user interface implementations for different languages and adapted to the resources available in the target devices. Such concepts have started to be adopted even in international standards, such as the W3C XForms. In some cases even reverse engineering transformations have been defined, which are able to build logical descriptions starting with existing implementations. This allows the reuse of content already available for the design of new user interface versions. Such transformations can be

Copyright is held by the author/owner(s).

CHI 2007, April 28 – May 3, 2007, San Jose, USA

ACM 1-xxxxxxxxxxxxxxxxxx.

incorporated in authoring tools (such as TERESA [7]) or can be part of software infrastructures able to dynamically generate user interfaces adapted to the device at hand during the user session. However, many issues have still to be solved in order to obtain general solutions for user interfaces in ubiquitous environments. Some of them are discussed in this position paper.

End User Development

One fundamental challenge for the coming years is to develop environments that allow people without particular background in programming to develop their own applications. Indeed, the increasing interactive capabilities of new devices have created the potential to overcome the traditional separation between end users and software developers. End User Development (EUD) [4] is a set of methods, tools, and techniques that allow people, who are non-professional developers, at some point to create or modify a software artefact. In this perspective, tasks that have traditionally been performed by professional software developers are transferred to the users, who need to be specifically supported in performing these tasks. New environments able to seamlessly move between using and programming (or customizing) can be designed. In the case of user interfaces obtained through transformations starting with logical descriptions the issue is how to allow end users to customize the transformation rules. This implies that even such rules be declaratively specified externally to the software performing the transformations. Such specifications should be represented and manipulated through representations that are understandable for end users. In addition, users should also be able to easily understand the impact of the modifications performed

on the transformation rules specifications on the resulting user interfaces.

The overall goal is to reach natural development, which implies that people should be able to work through familiar and immediately understandable representations that allow them to easily express and manipulate relevant concepts, and thereby create or modify interactive software artefacts. On the other hand, since a software artefact needs to be precisely specified in order to be implemented, there will still be the need for environments supporting transformations from intuitive and familiar representations into more precise, but more difficult to develop, descriptions. In this context Programming by Example PBE techniques can be useful. They have existed since 1975, the basic idea is to teach the computer new behaviour by demonstrating actions on concrete examples. An example of how to combine PBE and multi-device Web interface design is discussed in [5].

Generating User Interface for All

The research work in model-based design and in the accessibility area have some potential synergy, which has not yet been investigated. On the one hand, model based approaches aim to provide user interface descriptions where the logical function of each element or construct is highlighted removing useless implementation details, on the other hand disabled users often need support in order to orientate themselves. For example, when blind users access their Web pages through screen readers they can scan sequentially all the interface controls or all the links and so on. However, they often need to first have a general overview of the page content, which means a logical description indicating where is the heading, the

navigation bar, the search interface, the content part and so on. Accessibility aims to increase the number of users who can access a system, this means to remove any potential technical barrier that does not allow the user to access the information. Usability aims to make the user interaction more effective, efficient and satisfactory. Guidelines able to integrate accessibility and usability for specific classes of users, such as vision-impaired users [3] have been developed and specified in XML-based languages. They have been developed in such a way that they should indicate the user interface features that allow the information and the associated services, to be easily accessed by users with special needs. In the accessibility area some tools (for example MAGENTA [2]) able to check guidelines specified in XML externally to the tool implementation have been developed as well.

At this point, it would be interesting to obtain an integration of model-based transformations with accessibility and usability guidelines in such a way that the transformation rules be able to generate dynamically user interfaces depending on the target users by applying the set of guidelines that are more relevant for the current users and devices. Such transformations could be dynamically enriched by adding new guidelines expressed through the same guideline specification language without requiring changes in the implementation of the software tool supporting the transformations generating the user interfaces.

Generating User Interfaces for Dynamically Composed Web Services

Service-oriented architecture (SOA) is an architectural style whose goal is to achieve loose coupling among

interacting services. SOA enables dynamic, flexible applications which can always change rapidly by integrating new services as well as old legacy systems. Web service technology makes services available independently from the implementation of a particular application. There is a need for improvements in web service technologies regarding to the semantic description of service functionality, service operations and operation parameters, extending the descriptions in existing standards, such as web service description language (WSDL), in order to better support user interface generation.

One interesting research area is how to generate user-interfaces for dynamically composed web services. Generating user interface elements for services is the topic of several research efforts but there are still many open issues. For example, WSGUI deals with GUI inference from Web Service description, manually adding GUI hints to overcome limited self-information of services [8]. There is a need for more general solutions able to take Web service descriptions, which can be dynamically composed, and generate the corresponding user interface adapted to the context of use. This can be useful for example in domotic applications, in which middleware for domotic interoperability (such as Domonet [6]) can abstract a wide variety of domotic appliances (including media centers, security systems, light and temperature controls and so on), which may use various communication systems (Konnex, UpnP, BTicino, ...), and make their functionalities and states available through Web services. Such services dynamically depend on the domestic appliances available and enabled. At this point, a run-time environment should be able to dynamically generate user interfaces for

various possible interaction devices allowing users to control the appliances populating their home anywhere.

Migratory User Interfaces

Migratory user interface allow users to dynamically change device and continue task performance from the point they left off on the source device. While some research work has been dedicated to support migration from one to device to another one [1] or to partially migrate a user interface (for example for showing content on large screen and controlling interaction through a mobile device), there is the need for further research work to support migration when it involves multiple source and target devices and when it involves multi-user applications, in which each user can change device. In the case of multiple devices, there is a need to support coordination among them in such a way that the task distribution among them be as usable as possible and the state can be efficiently preserved when moving from one set of devices to another one.

Conclusions

While a number of XML-languages for supporting multi-device interfaces has been proposed, with associated tools, the complexity of ubiquitous environments still requires better solutions able to address the complexity of the relevant aspects in order to obtain usable and accessible solutions.

Some of the issues that need to be particularly considered are introduced and briefly discussed in this position paper. The overall goal is to obtain user interface description languages for ubiquitous environments, which allow for expressing and modelling relevant aspects in such environments:

interdependencies and configuration options of ubiquitous technologies, their dynamic behaviour, their privacy/visibility effects, their reliability aspects, together with device and environment descriptions and user-related aspects and their relationships

References

- [1] R.Bandelloni, G.Mori, F.Paternò, Dynamic Generation of Migratory Interfaces, Proceedings Mobile HCI 2005, ACM Press, pp.83-90, Salzburg, September 2005.
- [2] B.Leporini, F.Paternò, A.Scordia, Flexible Tool Support for Accessibility Evaluation, Interacting with Computers, Vol.18, N.5, 2006, pp.869-890, Elsevier.
- [3] B.Leporini, F.Paternò, Increasing Usability when Interacting through Screen Readers, International Journal Universal Access in the Information Society (UAIS), Springer Verlag, Vol.3, N.1, pp.57-70, 2004.
- [4] H.Lieberman, F.Paternò, W.Wulf (eds), End-User Development, Springer Verlag, ISBN-10 1-4020-4220-52006.
- [5] J. A. Macías, F. Paternò, Customization of Web applications through an intelligent environment exploiting logical interface descriptions, Interacting with Computers, In Press, Corrected Proof, Available online 6 August 2007.
- [6] Miori, V. Tarrini, L. Manca, M. Tolomei, G., An open standard solution for domotic interoperability, IEEE Transactions on Consumer Electronics, Volume: 52, Issue: 1, pp: 97- 103.
- [7] G. Mori, F. Paternò, C. Santoro, Design and Development of Multi-Device User Interfaces through Multiple Logical Descriptions, IEEE Transactions on Software Engineering, August 2004, Vol.30, N.8, pp.507-520, IEEE Press.
- [8] Spillner, J.; Braun, I.; Schill, A.: Flexible Human Service Interfaces, ICEIS 2007, accepted for publication