# Immediate User Interface Adaptation in Multi-device Environments

Sørensen, Henrik; Kjeldskov, Jesper

*Published in:*
Proceedings of PPD'12: Workshop on Infrastructure and Design Challenges of Coupled Display Visual Interfaces

*Publication date:*
2012

*Document Version*
Early version, also known as pre-print

# Immediate User Interface Adaptation in Multi-device Environments

Henrik Sørensen
Aalborg University, Department of Computer Science
Selma Lagerlöfs Vej 300, DK-9220
Aalborg, Denmark
hesor@cs.aau.dk

Jesper Kjeldskov
Aalborg University, Department of Computer Science
Selma Lagerlöfs Vej 300, DK-9220
Aalborg, Denmark
jesper@cs.aau.dk

## ABSTRACT
Taking full advantage of the wireless network infrastructure surrounding us, requires better interoperability between personal network capable electronic devices. In many cases it is restricted to data sharing or the inter-device interaction is cumbersome. One trend in research is the notion of plasticity, allowing user interfaces to adapt to various contexts and thereby support the user in the multi-device environments. We however argue that an important factor in the effortlessness of the user is the immediacy of the adaptation process. In this paper we explore the plastic interface types, distributed and migratory interfaces and propose proxemic interaction techniques as a framework to provide context awareness, which can help systems infer how and when to adapt.

## Categories and Subject Descriptors
H.5.2 [**User Interfaces**]: Graphical user interfaces (GUI)

## General Terms
Design

## Keywords
Plasticity, distributed interfaces, migratory interfaces, proxemic interaction, interaction design

## 1. INTRODUCTION
The establishment of a wireless network infrastructure surrounding us, has introduced easier connectivity between different personal electronic devices. In addition to enable data to be shared or synchronized between devices, it provides great potential for interaction design transcending the physical boundaries of devices such as desktop PCs, Laptops, smartphones, tablets, gaming consoles, TVs etc. Taking advantage of this potential is however challenging and a lot of issues are raised, when the user interface (UI) is no longer restricted to a single device.



An important aspect to take into account is the huge difference between enabling the same data to be accessed through similar interfaces on different devices and actually supporting interoperability from a user point-of-view. Take, e.g., the simple case of e-mail. In order to switch device in the middle of a reply, the user has to open the e-mail on the smartphone and begin the reply, save the draft, open the e-mail client on the laptop, locate and open the draft, finish the reply and send it. The data (the e-mail) is available, but the interaction becomes cumbersome because it has to be restarted on the new device. We argue that if we are to use the electronic devices seamlessly in a collaborative way, the UI has to be able to adapt to the context, and do so with a certain immediacy. This paper especially focuses on the potential of automation as means to create this immediacy, by removing some of the responsibility from the user in the interaction between different devices. Coutaz et al. describes the ability of an interactive system to adapt to variations of context while preserving usability, under the term plasticity [4]. They describe the plasticity and adaptation as a five step process going through detection, identification, selection, transition and execution. Each step can be initiated by either the system or user. Their work is however widely focused on the development process of plastic UIs.

Two related but distinct types of UIs are often considered in relation to plasticity: Distributed and migratory [1]. Each has a series of common and unique challenges, one being how to automate part of the five step process in order to achieve more seamless interoperability. In this paper we explore relevant differences between distributed and migratory interfaces by looking at examples and propose proxemic interaction techniques as a way of supporting immediate user interface adaptation.

## 2. DISTRIBUTED INTERFACES
The term distributed is used in various contexts with different definitions and meanings and similarly in HCI there is no clear definition of distributed interfaces. It basically describes an interface which is not restricted to a single device. Under the notion of plasticity the distributed interfaces are classified as mouldable, distributable and migratable [5]. Here we focus on the distributable interfaces which are not migratory. It is important to note that even though an interface is distributed but not migratory, it is not necessarily static. Adaptation can still be applied to individual interface parts distributed to separate devices and even to some

extent can interface parts move between devices. These interfaces are sometimes referred to as dynamic distributed UIs. Paterno et al. describes the difference between dynamic distributed UIs and distributed migration as:

> *"In distributing migration the interface migrates to multiple target devices, which is different from a distributed user interface, where the interface runs in one device and is allocated to multiple interaction resources connected to that device (for example, two screens)." [2]*

Although the term distributed interfaces itself implies some sort of inter-device dependency, an interface might be distributed in a way where the user interacts with the different parts individually. The real challenges are however introduced when the user has to use multiple devices together in, e.g., coupled display environments.

## 2.1 Case Study

In our previous work, the development and evaluation of a distributed multi-device, multi-user music system, called MEET, has been used to explore the interaction space surrounding distributed systems with co-located devices. In this example we consider the UI to illustrate the interaction design for the system. The concept is to allow co-located users to share their music, at e.g. a party, in order to nominate and vote for songs using their smartphones, thereby influencing the music in a collaborative manner. The interface is distributed and includes the following entities:

- Smartphone Application: This application is the primary input device for the music player. Besides the music sharing control, it features a nominate functionality, where users can browse the collection of music shared by users and nominate songs they would like to hear. It also features a vote functionality presenting a list of nominations that can be voted for. It is possible to give a positive or negative vote for each nomination, which will simply just add or subtract one point from the total score. An important aspect is to allow users to use their own smartphone and thereby make it represent the specific users choices which can be changed continuously.
- Common Tablet: The tablet application is basically a simplified version of the smartphone application, which can only be used for nominating and voting. It first of all serves the purpose of a public input device that can be used by people without a compatible smartphone and secondly to create a physical interaction point for the music system in general. Because the tablet is used by several users, the vote feature has been modified to include a 10 second countdown after a vote where the device is locked. This is done to prevent a single person from voting for a single song several times.
- Situated Display: Shows the primary visual output of the music player. It is designed for a large flat screen TV or projector and is placed with general visibility in mind.

Nominations are represented on the situated display as an album cover which are mapped to the mobile devices. On the situated display the current score is represented by size, meaning that the largest nominations are more likely to be played next. The voting interface is shown in Figure 1.


**Figure 1: The interaction devices of MEET.**

Specific parts of the interface are distributed onto specific devices each with their own output screen and each serving a specific purpose. There is no real notion of interface migration, as each device has very specific and static roles in the interaction design. The music system is still running in one place and interaction is merely distributed to other devices. There are however some interesting shortcomings of this static approach for interface distribution, which will be discussed later.

# 3. MIGRATORY INTERFACES

Grolaux et al. defines interface migration as:

> *"The migration of a UI is the action of transferring a UI from one device to another for example from a desktop computer to a handheld device." [7]*

The idea of application migration is not particularly new and an early attempt is, e.g., presented in [3]. Their approach is a general programming model based on an agent migration paradigm, meaning that actual program procedures are transferred along with the program state and compiled at a new device. A solution like this is however unsuitable for the kind of user interface migration enabled by the evolution of mobile devices and the incorporation of network capabilities into a variety of platforms.

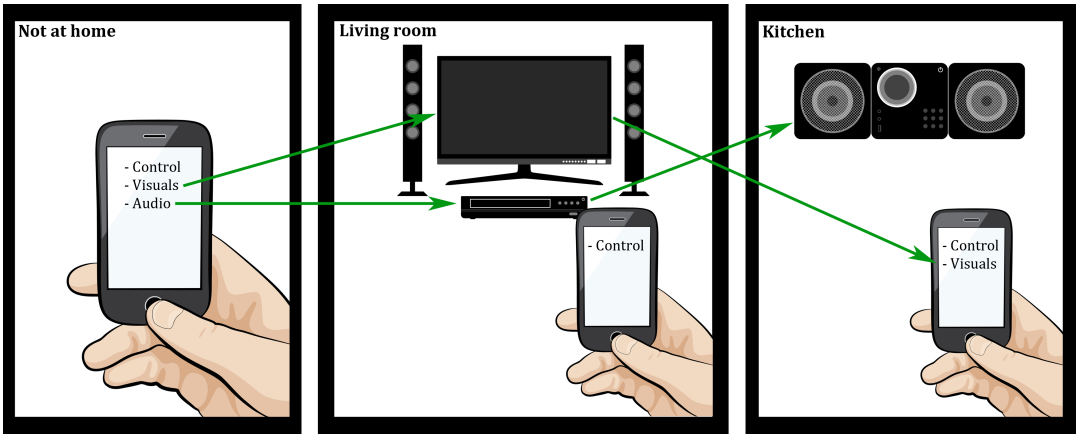


**Figure 2: The flow of interface migration.**

More recent research includes the work of Paterno et al. who have, among other things, formulated a taxonomy for migratory UIs [2]. They present basic concepts, a reference framework and 11 logical dimensions of interface migration. The dimensions describe very versatile aspects like how to activate the migration, the implementation environment and the architecture of migration platforms.

The strength of migratory interfaces is to enable the user to interact continuously with an application, even when a change of device is appropriate. In order to achieve this, the devices must be able to communicate seamlessly and the state of the application has to be preserved after transition.

## 3.1 Case Study

Following the music consumption case, we present an example of how user interface migration can fit into interaction design for related activities. The proposed design is first of all based on the utilization of a smartphone as a portable music player connecting to an online music library. However, when the user enters his home he has the option of partially migrating [2] the audio to his home stereo and/or the visuals to the flat screen TV, leaving the smartphone a remote control. The system is thus still controlled by the smartphone, which the user is already carrying, but browsing through the music library and the display of information about the song playing is moved to a much larger screen and decoding of the music along with the audio output is moved to a higher quality sound system. As the user enters different rooms he can choose to migrate the audio or video output to convenient devices. An example could be going to the kitchen to cook dinner, where no screen is available, migrate the audio output to a Wi-Fi radio and the visuals back to the smartphone. The flow of migration in this scenario can be seen in Figure 2.

In this example the music player on the smartphone is not just sending its video and audio output to other connected resources. Instead the actual interface parts are migrated to other devices, relieving the smartphone of both computational and network resource consumption. An important aspect of migration is maintaining the state of the music player when transitioning between devices, meaning that the user will be able to listen to the same songs continuously while preserving playlists and settings. The technological platform required to interface migration, as depicted in the case, is already present. There are however a wide range of challenges related to the interaction design of interface migration that needs to be considered.

# 4. PROXEMIC INTERACTION

According to [8] a lot can be determined about the interaction between people, from the interpersonal distance and orientation which led him to coin the term proxemics. In the work of Greenberg [6] the concept of these proxemics is extended by introducing five dimensions that additionally describes relations between different devices and between devices and users: Distance, orientation, movement, identity and location. The goal of the framework, is to support seamless interaction with multiple devices and reach a point where devices are aware of not only other devices present, but also the users and non-digital objects. Ultimately this will decrease effort from the user in terms of connectivity and configuration of devices and make them become more transparent in the everyday life. As part of the research, a proxemity toolkit has been developed which can help in the development of high fidelity prototypes capable of interpreting proxemic relations from sensory input [9].

## 4.1 Application

There are certain similarities in the objectives of proxemic interaction and interface plasticity, which are aimed towards seamless inter-device interaction in multi-device environments. Where plasticity is concerned with the ability to adapt UIs to differences in contexts, proxemic interaction rather provides the means to obtain context awareness, which can in turn aid in the decisions of interface adaptation. Considering UI adaptability in the first place is a big step in the right direction. A challenge is however to make the adaptations meaningful and easily accessible, which is why we emphasize the immediacy of the process. Even when migration is possible, designing the migration control and activation is still a non-trivial task. In the following we explore how proxemics can add value to the example case studies of distributed and migratory interfaces, as well as some of the introduced challenges.

### 4.1.1 Distributed Interfaces

Results from a field study of the distributed UI of MEET showed the importance of the location of users and devices in the physical environment. In cases where the user had great visibility of the situated display, the most important property was the coordination of visual feedback between the situated display and the smartphone application. However when the users were either at a distance, or otherwise unable to clearly see the display, they would be highly dependent on the limited feedback given from the smartphone application. It could be argued that a redesign of the interface or added features would solve this problem, but an important aspect of the smartphone application is also the simplicity as the users were engaged in a social activity as well. Proxemic interaction techniques could in this case be applied to infer when a user lost visibility of the situated display and provide additional feedback on the smartphone application. Because the system is keeping track of the user, immediate adaptations are made possible, relieving the user from an overhead of accessing different views or changing settings whenever his position changes.

### 4.1.2 Migratory Interfaces

In the case of a migratory music setup, the use of proxemics is more apparent. One of the challenges, in the user interaction of migratory interfaces, is how to activate the migration process, which can be either on demand or automatically [2]. As a lot of the work in this area is concerned about how to handle the migration backend architecture, few solutions are considering how to enable the system to infer when to migrate and whether it is appropriate to migrate completely or partially. Proxemic interaction could be used as a framework to keep track of spatial relations between the user and devices, thereby inferring when migration is possible and determining the most convenient way of doing it. Once more the power of automation is to require minimum effort from the user in the plasticity and adaptation process.

### 4.1.3 Challenges

A major challenge of the utilization of proxemics in both distributed and migratory interfaces is the diversity of devices and physical environment. Distances can be interpreted differently in different size rooms with different size devices, which in turn can influence which device is appropriate. Hall's notion of proxemics are fixed zones determined by measurable distances ([8]). The same can be difficult to achieve when talking about proxemic interaction, as the devices vary on several parameters. There are of course differences between people as well, but they appear to be less significant, in this context, as is the case for electronic devices.

Another related challenge is the accuracy of proxemic information. A pragmatic approach could be to simplify the proxemic framework and focus on specific dimensions like distance and orientation. Greenbergs proximity toolkit could be an option as a platform for collecting sensory input and the support for the Microsoft Kinect could provide an easy setup for a prototype development. The idea would be to have a central server keeping track of proxemic information and let the clients at the different devices adapt through pre-compiled interfaces.

## 5. CONCLUSION AND FUTURE WORK

What we have explored is the relationship between different types of plastic UIs, namely distributed and migratory, and suggested proxemic interaction as a framework which can be used to support immediate interface adaptation. The emphasis is put on the immediacy as the ability to adapt in itself does not necessarily provide value to the interaction in every situation. It is important to note that immediacy is not only a matter of network response times and computational power. It is also a question of how to design the interaction to support the user.

There are differences to the design of distributed and migratory interfaces. In terms of the plasticity and adaptation process there are however certain similarities, which in both cases can benefit from work on automation. To gain insight into the utilization of proxemics in UI adaptation, future work will consist of the development and evaluation of prototypes, e.g., built as an implementation of the Proximity Toolkit [9].

## 6. REFERENCES


[1] L. Balme, R. Demeure, N. Barralon, J. Coutaz, G. Calvary, and U. J. Fourier. Cameleon-rt: A software architecture reference model for distributed, migratable, and plastic user interfaces. In *In EUSAI*, pages 291–302. Springer-Verlag, 2004.

[2] S. Berti, F. Paternó, and C. Santoro. A taxonomy for migratory user interfaces. In *Interactive Systems. Design, Specification, and Verification*, Lecture Notes in Computer Science, pages 149–160. Springer Berlin / Heidelberg, 2006.

[3] K. A. Bharat and L. Cardelli. Migratory applications. In *Proceedings of the 8th annual ACM symposium on User interface and software technology*, UIST '95, pages 132–142, New York, NY, USA, 1995. ACM.

[4] G. Calvary, J. Coutaz, and D. Thevenin. A unifying reference framework for the development of plastic user interfaces. pages 173–192. Springer-Verlag, 2001.

[5] A. Demeure, G. Calvary, J.-S. Sottet, and J. Vanderdonkt. A reference model for distributed user interfaces. In *Proceedings of the 4th international workshop on Task models and diagrams*, TAMODIA '05, pages 79–86, New York, NY, USA, 2005. ACM.

[6] S. Greenberg, N. Marquardt, T. Ballendat, R. Diaz-Marino, and M. Wang. Proxemic interactions: the new ubicomp? *interactions*, 18:42–50, January 2011.

[7] D. Grolaux, P. Van Roy, and J. Vanderdonckt. Migratable user interfaces: beyond migratory interfaces. In *Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004. The First Annual International Conference on*, pages 422 – 430, aug. 2004.

[8] E. T. Hall. *The Hidden Dimension*. Doubleday, 1966.

[9] N. Marquardt, R. Diaz-Marino, S. Boring, and S. Greenberg. The proximity toolkit: prototyping proxemic interactions in ubiquitous computing ecologies. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, pages 315–326, New York, NY, USA, 2011. ACM.