Multiple User Interfaces: Why Consistency is Not Everything, and Seamless Task Migration is Key

Pardha S. Pyla, Manas Tungare, Manuel A. Pérez-Quiñones
Dept. of Computer Science and Center for Human-Computer Interaction
Virginia Tech
Blacksburg VA USA.
{ppyla, manas, perez}@vt.edu

ABSTRACT

In this position paper, we argue that in the context of a user's interaction with multiple platforms, consistency must be supported along with support for user's task migration. We believe that the user should be able to perform a task using multiple user interfaces through an application that may or may not be similar at the interface level on each platform but, more importantly, one that supports seamless task migration. Further, we believe that consistency is desirable as long as it supports seamless task migration but should not dictate the design. We present examples of how successful multiple user interfaces have tackled the issue of consistency and discuss examples of why we should support task migration as a higher goal than consistency. We present a definition of task disconnect and its relationship to task migration. We also discuss some design issues related to consistency and task disconnect that must be considered in the development of multi-platform user interfaces.

Author Keywords

Multiple user interfaces, consistency, continuity, continuous user interfaces, task disconnect.

ACM Classification Keywords

H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION

In this age of mobile computing, it is extremely common for users to perform their tasks using multiple devices ranging from the traditional desktop to the now ubiquitous cell phone. Many applications are available on handheld devices, and this fact illustrates the demand from users to be able to perform work-related duties while mobile.

When designing interfaces for multiple devices, emphasis is placed on maintaining consistency so that users may be able to transfer their knowledge of existing systems to newer devices. It is not clear, however, what consistency means in this context. Should the focus of consistency be at the interface level (e.g. its appearance and behavior; look and feel), the task level (e.g. support for the same tasks but with different look and feel), or simply the data level (e.g. read/write same file formats)?

A scenario commonly used to illustrate the Multi-Platform User Interface concept (MUI, or MPUI) [12, 17] highlight what we call task disconnect when using multiple user interfaces. The scenario describes a user, while on the road, needing to access a file on the laptop that is in the trunk of the car. The file cannot be automatically accessed from other devices, instead it requires the user to proactively obtain the file first and then use it. Thus, although the data is available on an alternate platform, the application has not been designed to account for task migration and thus cannot handle the data management itself. In this example, there is less-than-adequate support for seamless task migration, placing the burden of this activity on the user. Situations like this often require the user to be pro-active in planning their task migrations ahead of time. The user is often required to copy, sync, and/or duplicate data so that it is available in multiple devices for future situations.

In the rest of the paper, we argue that consistency needs to be better defined if it is to be the overriding factor in the design of multiple user interfaces. We believe that the division of tasks across devices, contexts of use, device affordances, and intra-platform consistency must be carefully considered when defining and designing for MUI consistency.

DEFINITION: TASK DISCONNECT

The need to transfer task information back and forth between various platforms burdens users with methods like USB key drives, remote desktop software, e-mail, network file storage, and many other means. These attempts create a gap in task execution. At a high level, we define this gap as a *task disconnect*.

Qualitatively, a task disconnect represents the break in continuity that occurs due to the extra actions outside the task at hand that are necessary when a user attempts to accomplish a task using more than one device. This disconnect occurs because moving a task from one primary device to a secondary device requires stopping work, transferring current data and files to the secondary device, opening and loading an assortment of applications on the secondary device to complement or replace the applications being used on the primary device, and then opening the information and data with the secondary device's loaded applications to restart work on the original task.

Task disconnects can also be envisioned as analogous to interruptions, but occurring over multiple platforms, locations, contexts, and most importantly, over a much larger time interval (e.g. a user works on a PDA on the field and then moves to an office after a day or two to work on the same task on a desktop). Interruptions are events that break the user's attention on a particular task to cater to another task that is in need of attention, and are the focus of a whole area of study by themselves [9, 14]. The issues in these two areas of interruptions and task disconnect research are at some level similar: how to help the user make a switch from one task condition to another in such a way that the user's cognition, attention, reaction, physical and memory loads are respected.

However the research objectives of these two seemingly related areas are widely different. The research on interruptions focuses on striking a balance between the needs of the interruption and the disruption resulting thereof. Our research on task disconnects attempts to use continuous user interfaces (CUIs) to support the affects of an unavoidable interruption of a task across platforms in such a way that the costs of recovery are minimal. We believe that *consistency in interface design should be followed only when it helps reduce the cost of task disconnects and helps support seamless task migration.*

PREVIOUS WORK

A review of the MPUI literature shows a few studies that have tried to address the problem of migrating tasks or applications over multiple platforms. However, most of these studies have focused primarily on the technological aspects of this problem. For example, Chu et al. [8] take the approach of migrating an entire application to support seamless task roaming . However, their approach has considerable latency during migration (interrupting the user's task sequence) and does not discuss the implications on the user's tasks and goals.

Similarly, Bandelloni and Paternò [2] talk about user interaction with an application while moving from one device to another. They describe three levels of migration: total, partial and mixed. Chhatpar and Pérez-Quiñones [7] call this migration "dialogue mobility" and propose a requirement for the application data and logic to be separate from the user interface, but neither one of these projects take the task perspective we propose in this paper.

Florins and Vanderdonckt [11] describe rules and transformations that attempt to provide graceful degradation of user interfaces as the application is migrated from one platform to another. Even though their work is based on the same principle of continuity between devices from an interaction perspective, their focus is on the user interface generation and not on task migration.

From a systems perspective, toolkits and tools such as TERESA [15] and ARIS [3] have utility in rapidly deploying applications that can be migrated over multiple platforms, but do not address the task semantics that users wrestle with while trying to interact with an MPUI.

Often, people interact with a computing platform and later find the need to access the information they came across but from a different platform. Most software systems do not support this requirement. One of the few exceptions where user interfaces actually support such re-finding [5] of information across platforms is WebContext [4], a voice-based phone user interface with which users can re-find information they have previously seen on the web while at their desktops.

DESIGN ISSUES TO CONSIDER

How can we maintain task continuity across multiple devices? Task continuity requires interfaces that support the transfer and recovery of state and activity context. Recovery of activity context deals with the ability to recover the last few actions that were performed on one device so that they can be taken into account while migrating the task to another device. The goal of our research is to define a *continuous interaction paradigm* and to understand how we can provide seamless transfer of information and tasks across multiple devices to reduce the effect of task disconnects. We refer to interfaces in this paradigm as *continuous user interfaces*.

Seamless migration is partially dependent on knowledge continuity and task continuity [10]. Knowledge continuity requires visual continuity, both graphical and textual, successful partitioning of data and functionality, and procedural consistency. Partitioning of data and functionality deals with how a program divides what functions and what data is most appropriate on each device. Having a desktop calendar application show the entire month as a first view with overview information for each day put on the screen simultaneously is reasonable. On a cell phone, it is more appropriate to show only today's activities. This is an example of data partitioning and is sometimes at odds with requiring all platforms to be consistent.

On the other hand, one could argue for consistency and that this sort of variation of the data and functionality to fit the capabilities and factors of each platform may be in contradiction to the users' mental model of the system. So a question that needs to be explored is: Will users build a mental model of an application domain on the first device they use? Will users have difficulty in adapting these mental models to a new and different platform without difficulty? We cannot answer this question with certainty, yet recent results [6, 16] show that users can adapt their mental models to different devices without much difficulty. So, it might be more important to achieve consistency at the mental model and data model level rather than the interaction level.

Based on our experiences and observations so far, we propose that interaction designers must present to the user an application that supports seamless task migration. To move towards this goal, we suggest a few general guidelines.

Holistic Interaction Design

When designing for an ecosystem of devices, it is necessary to consider all platforms together and distribute or replicate functionality according to the affordances and contexts of use of each device. This may require forfeiting interface-level consistency between two or more platforms in favor of presenting a 'holistic' interface to the user. A holistic design approach would eliminate issues like when a user is sitting at a desk in an office, and a PDA, cell phone, and desktop computer all sound an alarm at the same time because each device is independent of the rest. In spite of the calendar application(s) on all three devices being mutually consistent, the devices demonstrate a lack of awareness of the presence of other devices within the user's workspace. Ideally, the design of these devices should take a task-oriented approach and issue a single alarm to the user on the current dominant platform, as determined by the devices themselves.

iPod + iTunes: Task Distribution

As an example of the distribution of tasks across devices, we consider the popular iPod + iTunes [1] multi-platform interface. The iTunes application supports media management creating and editing playlists, music playback, access to the iTunes Music Store and importing music from CDs. iTunes provides this functionality through a desktop-sized multi-pane user interface. The iPod has its own music-browsing interface on a 2-inch screen that can be operated using a touch-sensitive scroll-wheel and a few buttons. The iPod has a simple browsing mechanism specifically tailored to selecting music for playback purposes only. Both these platforms leverage the affordances that are unique to the device to support the 'holistic' task of music management and playback.

The music-browsing interface on the iPod is 'consistent' with iTunes to the extent that they both use the same set of playlists, music organization and media files. It does not, however, provide the same functionality that the iTunes application provides. This does not adversely affect the user's ability to listen to music because consistency is not the overriding concern in this example; instead, complementarity of tasks is. The iPod + iTunes is an excellent example of an application domain where tasks are distributed among devices in complementary ways. Furthermore, the iPod + iTunes dual-platform interface enables seamless task migration between the two devices, since synchronizing media files is a one-way process (from iTunes to iPod) and requires almost no interaction; files are automatically synchronized upon plugging in the iPod.

Workflow Adaptation

It is common today to find users moving their work back and forth between different computers (e.g. home and office). The need to transfer task information between various platforms burdens users with methods like USB key drives, remote desktop software, e-mail, network file storage, and many other means.

An example of people adapting their workflow to try and mitigate a task disconnect is when they use alternate strategies to try to keep the work products of an interaction available to themselves. Jones, Bruce and Dumais [13] found that people often emailed the URLs of the websites they visited or the 'favorites' list to themselves when they have the need to access it from a different location. This approach is often preferred over bookmarking at the browser level since most bookmarks are tied to each individual browser. Many solutions have emerged to compensate for this problem, from commercial services like .Mac to social bookmarking services like del.icio.us.

In this example, users of multiple platforms adapted their workflow to compensate for the lack of support of task and/or data migration. The available option (e.g. bookmarking) does not support migration, thus users included in their workflow emailing their work to themselves. It is worth noting that this problem is much larger than just for bookmarks. We have observed that many mobile workers have opted for using a single computer, their laptop, instead of moving files back and forth between their office and their off-site locations. To minimize the task discontinuity, users simply opt to 'carry their office with them' by keeping all of their data in a single device (laptop) and taking it with them.

Migrating Task State and Data

To achieve seamless task migration, some or all of the data associated with a running application needs to be transferred from one device to another before interaction can proceed seamlessly on the second device.

Some part of the user data is clearly *semi-public*, e.g. calendar information is often distinguishable into a private portion and a public portion. Sharing the public portion with colleagues makes it easier to schedule meetings, etc. However, this shared view is not always an exact copy of the data available on the user's primary machine. The specifics of events, for example, can be 'blurred out' by marking certain times as *free* or *busy* without providing any more details. In this case, the public view of the user's calendar is not completely consistent with the private view, and this compromise in consistency is necessary to maintain the required level of privacy in the application.

Given the vastly different capabilities of various devices, it is not always practical to transfer the entire task data from one device to another. Shared data across platforms requires at best partial consistency. For example, synchronization software currently available on Mac OS X copies only a few upcoming weeks at a time to cellphones. The software assumes that the devices will be synchronized frequently (at least weekly) and thus saves memory in the cellphone by only having upcoming events stored. We believe that the context of interaction, the capabilities of the device, and the appropriateness of the interaction are more important in designing MPUIs than consistency alone.

Intra-Platform Consistency

The application developed for each platform must stay consistent with design guidelines for that particular platform. Apple's design guidelines for Mac OS applications state that the default button in a dialog box ought to be the right-most in a row of buttons, whereas Microsoft's design guidelines for Windows applications suggest that it should be the left-most. Most applications available for both platforms seem to prefer platform consistency, thus leveraging the user's familiarity with the platform, rather than with the application.

CONCLUSION

From the above discussion, we reaffirm our position that seamless task migration across multiple platforms is a more important concern than simple consistency of interfaces across platforms. Task disconnects that occur when suspending a task on a device and resuming it on a second device can be mitigated by using interaction techniques that leverage the unique capabilities of each platform, instead of trying to replicate everything across all platforms in the name of consistency. The process of interaction design for each platform cannot be performed in isolation, or with the sole objective of staying consistent with the interface on other platforms, but a holistic view is needed. Task state and data must be partitioned according to the affordances of the platforms in question and not on the basis of consistency alone. Applications developed for a particular platform should respect the design guidelines stipulated for that operating environment.

At Virginia Tech, we have been working in this research area for the past couple of years. We have theoretically defined concepts such as *task disconnect* and are attempting to understand the measures for seamlessness in task migration. We created the concept of *Continuous User Interfaces* to study the problem of task disconnect in Multiple User Interfaces. Although much work remains, we believe that we are in a position to make an impact in the design of user interfaces for multiple platforms.

REFERENCES

1. Apple, Inc. iPod. http://www.apple.com/ipod/, Last Accessed: December 2005.
2. R. Bandelloni and F. Paternò. Flexible interface migration. In *Proc. 9th International conference on intelligent user interface (IUI)*, pages 148–155, 2004.
3. J. Biehl and B. Bailey. ARIS: An interface for application relocation in an interactive space. In *Proc. 2004 conference on graphics interface*, pages 107–116, 2004.
4. R. G. Capra, M. Pérez-Quiñones, and N. Ramakrishnan. Webcontext: Remote access to shared context. In *Proc. Perceptual User Interfaces Workshop (PUI 2001)*, 2001.
5. R. G. Capra and M. A. Pérez-Quiñones. Using web search engines to find and refind information. *IEEE Computer*, 38(10):36–42, October 2005.
6. C. Chhatpar, S. Lambros, and M. A. Pérez-Quiñones. Variability of user interaction with multi-platform news feeds. Technical Report TR 04-22, Dept. of Computer Science, Virginia Tech, 2004.
7. C. Chhatpar and M. Pérez-Quiñones. Dialogue mobility across devices. In *Proc. ACM Southeast Conference (ACMSE)*, 2003.
8. H. Chu, H. Song, C. Wong, S. Kurakake, and M. Katagiri. Roam, a seamless application framework. *The Journal of Systems and Software*, 69:209–226, 2004.
9. M. Czerwinski, E. Horvitz, and S. Wilhite. A diary study of task switching and interruptions. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 175–182, New York, NY, USA, 2004. ACM Press.
10. C. Denis and L. Karsenty. *Inter-Usability of Multi-Device Systems - A Conceptual Framework*, pages 373–384. Multiple User Interfaces: Cross-Platform Applications and Context-Aware Interfaces. John Wiley and Sons, 2004.
11. M. Florins and J. Vanderdonckt. Graceful degradation of user interfaces as a design method for multiplatform systems. In *Proc. 9th international conference on Intelligent user interface*, pages 140–147, 2004.
12. R. Ghani. 3G: 2B or not 2B? the potential for 3G and whether it will be used to its full advantage. *IBM Developer Works: Wireless Articles*, August 2001.
13. W. Jones, H. Bruce, and S. Dumais. Keeping found things found on the web. In *Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM 2001)*, 2001.
14. D. C. McFarlane and K. A. Latorella. The scope and importance of human interruption in human-computer interaction design. *Human-Computer Interaction*, 17(1):1–61, 2002.
15. G. Mori, F. Paternò, and C. Santoro. Tool support for designing nomadic applications. In *Proc. 8th international conference on intelligent user interfaces (IUI)*, pages 141–148, 2003.
16. A. Parush. The measurement and comparison of cross-device mental models. http://www.carleton.ca/-hotlab/hottopics/Articles/OnMentalModels.html, 2004.
17. A. Seffah and H. Javahery, editors. *Multiple User Interfaces: Cross-Platform Applications and Context-Aware Interfaces*. John Wiley and Sons, Ltd., 2001.