Continuous UIs for Seamless Task Migration in Multi-Platform UIs: Bridging Task Disconnects

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

ABSTRACT

The proliferation of mobile computing has changed the work environment forever, and information workers increasingly use multiple devices to achieve their tasks. As a consequence, users are forced to interrupt their tasks, move their data and information back and forth among the various platforms manually to accomplish their tasks and then resume them on another device. They trudge out USB key drives, remote desktop software, e-mail and network file storage in an attempt to mitigate the adverse effects of task migration which they must perform themselves. We refer to this break from the task at hand as a *task disconnect*. A task disconnect represents the break in continuity that occurs when a user attempts to accomplish his or her tasks using more than one device. Our objective is to study how software can bridge this task disconnect, enabling users to seamlessly transition their tasks from one device to another. We call the user interfaces to such software systems *continuous user interfaces*. We present the theory, definition, and discussion of task disconnects, our approach towards bridging this disconnect, and a prototype of a continuous user interface that was built to be used across the desktop computer and the tablet computer platforms. We then describe our subjective evaluation to measure the effectiveness of the prototype in bridging task disconnects and ensuring *seamless task migration*. We conclude with the results and insights gained from our evaluation.

Author Keywords

Continuous User Interfaces, Task Disconnects, Seamless Task Migration.

ACM Classification Keywords

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

INTRODUCTION AND MOTIVATION

In this age of mobile computing, it is extremely common for users to perform their tasks using multiple devices such as desktop computers, laptop computers, personal digital assistants (PDAs), cell phones and other such devices. Two or more of these devices are used either simultaneously or one after the other to achieve a single task. Many applications have been directly ported to handheld devices, and this fact illustrates the demand from users to be able to perform work-related duties when mobile.

The massive storage and computational power of the desktop computer has helped it to continue to be a central part of our daily work. From our own surveys, the desktop computer and the laptop computer are the two primary devices that people synchronously and simultaneously use to accomplish their daily work. This usage of multiple devices to accomplish a single task is the source of contention. Consider the following scenario.

Scenario

Amy is a graduate student working on a presentation for her biology class. While using her laptop computer in the library to collect images, references, and to take notes from a few journals in her research area, she sees a call for papers from one of the journals with a deadline coming up in the next few days. She makes an entry in a calendar program on her laptop to remind her the next day to submit an abstract to that journal. After finishing her work at the library, Amy returns to her office to finish the presentation on her desktop computer. She connects a USB key drive to her laptop to copy the files that she collected and created at the library to her desktop to incorporate into her presentation.

As she creates the presentation, she remembers a paper she had read a few weeks ago that could provide some background information for her topic. She tries to remember where she had saved that paper. She searches among her files on the desktop and realizing that it is not there, looks for it on the laptop. She finally locates that file and uses the USB key drive again to transfer it to her desktop.

She resumes working on the presentation and after a couple of hours, finally finishes it. While she is working, she makes some changes to the spreadsheet she created in the library. Because the version of the spreadsheet on her laptop is now out of date, she uses her USB drive again to update that file on her laptop. She leaves for home, happy that she is done with the presentation, completely forgetting about the “Call for Papers” calendar event on her laptop.

Why must Amy be forced to manage this interaction herself? The amount of duplicate effort in this scenario clearly shows that the burden of transferring information and accomplishing a task using multiple devices jointly is being placed on the user. The situation in which Amy is being forced to drag through the plethora of devices, moving files back and forth, opening and closing applications and repeatedly copying and pasting information is what we describe as a *task disconnect*.

BACKGROUND AND THEORY

Before formally defining a *task disconnect*, we first describe tasks and the various parameters of tasks.

Tasks, Activities, Units and Cost

A task can be defined as “a goal to be attained in given conditions” [16]. These conditions can be expressed using three points of view: “the states to be covered, the permitted operations, and the procedure” [17]. At a slightly lower level, tasks can be said to be composed of activities. An activity is “what the subject puts into operation (cognitive operations, behavior) in order to meet task demands” [17]. We also make use of Leplat’s definition of *elementary units* to be the “elementary tasks, and elementary states or operations.” Leplat uses these definitions to describe task complexity. However, we use the term *units* to further subdivide activities to their lowest granularity.

This decomposition of tasks into lower level units is somewhat similar to the description of goals and operators in the GOMS (Goals, Operators, Methods and Selection rules) model [8,15], where a task loosely corresponds to a goal and a unit to an operator. Only, we define an intermediate level of decomposition called activity to provide for a finer level of analysis. (In the GOMS model these are referred to as sub-goals.)

For nontrivial tasks (i.e., tasks that involve multiple activities), we define a *procedure* to be an operation execution sequence of multiple units. We also associate a parameter required for the successful execution of a unit: an *instruction*. Instructions are knowledge directions necessary to execute units, that can exist in the user’s understanding of the world or it can exist in the aids and artifacts in the task environment.

A parameter that is fundamental to each unit is *cost*. Cost is a multidimensional attribute set that is incurred during the execution of a unit [17]. These dimensions could be cognitive, physical, memory-intensive, resource-intensive or a combination depending on the nature of the unit and the expertise of the user.

Another important parameter of a task is time. In the words of Leplat, “every task takes place in time and may be described by the temporal dimensions of its organization”. Out of the few temporal dimensions that Leplat describes, *temporal ruptures* is of particular importance to our work. We adapt and modify Leplat’s definition of temporal ruptures to mean interruptions by activities that do not directly contribute to the successful execution of the task at hand.

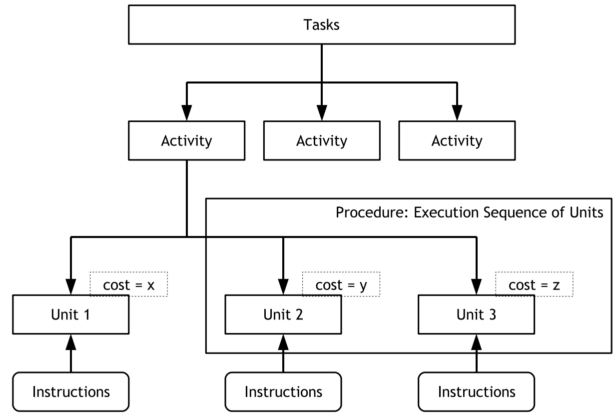

Figure 1. Tasks, Activities, Units and Instructions

Task Disconnects

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.

Theoretically, we define a task disconnect to be *a temporal task rupture arising due to activities required to manipulate multiple devices which are used to accomplish the task, but which do not directly aid in the completion of the task at hand; i.e. which are not directly included in the task procedure*.

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 [19–21]. 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.

Another field of study that talks extensively about interruptions is Linguistics [3,13]. Even though there is considerable debate as to what exactly constitutes an interruption in this domain, one can safely say that interruptions occur often in normal conversations. However, in this field of study, not all interruptions are disruptive or need repair. Even in cases where the interruptions are disruptive, the costs associated with repair are low. This is because humans have an inherent ability to repair, recover and proceed with most of the conversations using their ingrained social and cultural aids.

However, the research objectives of these two seemingly related problems of interruptions and task disconnects appear to be widely different. The research on notification systems focuses on striking a balance between the benefits of notifying by interruption (using software or physical interfaces) and the disruption resulting thereof, on a given platform. 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.

RESEARCH QUESTIONS

So what does it mean for a task to not be disconnected? In other words, 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.

Our goal in this research is to address these questions of providing a continuous interaction paradigm and to understand how we can provide seamless transfer of information and tasks across multiple devices to prevent task disconnects.

We characterize such seamless migration to be dependent on knowledge continuity and task continuity [12]. Knowledge continuity requires visual continuity, both graphical and textual, successful partitioning of data and functionality, and procedural consistency. Visual continuity identifies the fact that small changes in a program's visual features, the way things are laid out, the wording that an application uses, and the spatial orientation of various pieces of information, all have an effect on the usability of that program. Poor usability implies that the time to transfer productivity between the devices will be affected.

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 PDA, a small monthly calendar with the ability to select a week and see the information for that week is more appropriate. Data partitioning leads to potential inconsistency between interfaces on different platforms, and raises the question of whether this sort of adaptation of the data and functions to fit the capabilities and factors of each platform may be in contradiction to the users' mental model of the system on a single platform. Will the users build a mental model of the system on the first device they use? Will the users adapt those mental models to a new and different platform without difficulty? Even though one cannot answer this question with certainty, recent research seems to point that way [9, 24].

With the theory and background described above, we state our research question: How can we construct user interfaces that support a seamless transition for a user attempting to complete a task with more than one device, bridging the task disconnect that occurs during the transition?

To explore this question, we constructed a prototype of what we envision to be a CUI that specifically accounts for knowledge and task continuity to seamlessly bridge task disconnects and subjectively measure the perceived efficiency between using the prototype and traditional disconnected applications' UIs when attempting to accomplish a task across multiple devices.

RELATED WORK

The use of secondary computational devices is not necessarily in isolation from traditional forms of computational devices. Bellotti and Bly observed information workers to be mobile within the confines of their office [2]; this *local mobility* existed mainly to enable the use of shared resources and for communication with other staff members. From our own surveys, the desktop computer and the laptop are the two primary devices that people synchronously and simultaneously use to accomplish their daily work.

Our work has a strong parallel to the traditional Computer Supported Cooperative Work (CSCW) discipline [23]. Whereas, CSCW researchers focus on and attempt to have a seamless interaction between multiple users across space, time, distance and location in a collaborative setting, our objective is to provide a seamless interaction between multiple devices for a single user in the context of an execution of a task across time and distance.

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. take the approach of migrating an entire application to support seamless task roaming [11]. However, their approach has considerable latency during migration (interrupting the user's tasks sequence) and does not discuss the implications on the user's tasks and goals.

Similarly, Bandelloni and Paterno talk about user interaction with an application while moving from one device to another [1]. They describe three levels of migration: total, partial and mixed. The criterion the authors use to distinguish these three levels is based on whether user interaction (control part) or the information presentation (visualization part) is moved between the various platforms. Chhatpar and Pérez-Quiñones call this migration "dialogue mobility" and propose a requirement for the application data and logic to be separate from the user interface [10]. Neither one of these projects take the task perspective we propose in this paper.

Florins and Vanderdonckt describe rules and transformations that attempt to provide graceful degradation of user interfaces as the application is migrated from one platform to another [14]. The objective of their work is to maintain continuity between devices from an interaction perspective. Even though their work is based on the same principle of continuity, their focus is on the user interface generation and not on task migration.

Biehl and Bailey introduce ARIS [4], a window management framework to relocate running applications from one display to another. Mori et al. [22] describe a tool called TERESA that helps in designing and developing model-based nomadic applications. Toolkits and tools such as TERESA 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 during that interaction. Most software systems do not support this kind of requirements. One of the few exceptions where user interfaces actually support such re-finding of information [7] is the WebContext system. Capra, Pérez-Quiñones and Ramakrishnan describe WebContext [5, 6], a voice-based remote access system with which users can re-find the information they have previously looked at on their desktops.

Another example of a system people use in everyday life to bridge the disconnect between their need for remembering information and actually being able to access it at anytime is the commonly scribbled notes on bits of paper. People write notes on bits of papers and carry them around or use them as reminders by sticking them on a refrigerator door or on a desktop screen. In this case, paper is the most cost-effective solution to bridging the tasks in everyday life. Lin, Lutters and Kim call these ‘micronotes’ [18]. Micronotes can be categorized as a specialized case of CUIs on a single platform with a simple input mechanism.

Denis and Karsenty provide a conceptual framework for “inter-usability” of multiple devices [12]. They provide an analysis of different cognitive processes in inter-device transitions and postulate two dimensions required for seamless interaction: knowledge continuity and task continuity. We base our work and the definition of CUIs on this requirement for seamlessness. We take this task-centered approach to solving the problem and we provide a definition, description, parameters, requirements, and prototype to demonstrate a seamless interaction over multiple platforms without task disconnects.

SAMPLE APPLICATION DOMAIN

We targeted a specific application domain with sufficient complexity to allow us to observe clearly the different parameters responsible for task disconnects. Because of the software engineering background of the team members, our choice of application domain was software development. Most specifically, we chose to build a prototype to support the preliminary design phase of software engineering where developers must collect customer requirements and generate initial design prototypes, diagrams, and models. We chose this application domain because of the need to use several tools such as text editors, drawing packages, scheduling programs, etc. when accomplishing a task, and because the nature of the task requires the use of multiple devices (interacting with customers and sketching requires some level of mobility). The other advantage with this choice of application domain is that we have immediate access to a qualified participant pool in this domain to evaluate our work.

USER SURVEYS AND INFORMAL INTERVIEWS

We used informal interviews and user surveys to gather insights into an example task of prototyping and the existence of disconnects when using multiple devices to prototype. We interviewed a total of 6 professional software developers. We asked open-ended questions targeting the technologies and devices they used to prototype and any insights into disconnects arising due to the mediation by these technologies. In addition, we also developed an online questionnaire that targeted software developers, graduate students with software development experience, and researchers in HCI who are familiar with computing and do prototyping tasks. The results of the analysis of the 32 responses are summarized in the Survey Results section.

SURVEY RESULTS

Our first group of questions was targeted at determining the usage patterns and platform preferences of users in the day to day tasks. For both personal and work related usage, the desktop overwhelmingly turned out to be the primary device of choice Laptop computers were the most popular complementary devices other than the desktop.

The two devices that are used most for prototyping tasks seem to be the pen-and-paper and a laptop. The obvious inference here is that people use multiple devices such as pen-and-paper and laptops when they have to perform a task being away from their desktop. Because of the collaborative and distributed nature of prototyping tasks, we were interested in the types of devices people commonly used to collaborate and distribute the artifacts generated after the prototype sessions with clients. Out of the 32 surveyed, 19 answered that they share the pen-and-paper sketches physically, whereas 18 said they used laptops and 17 said they used desktops to share. We infer that a large group of users actually digitize their sketches as they are (scan paper documents) or into high-fidelity prototypes (use drawing tools, e.g. Visio) if they have to transfer and share as they selected desktop and laptops as the devices for sharing. Another observation that supported this inference is that a majority of 24 people answered that they used email and 11 people used network file sharing, both of which require converting pen-and-paper artifacts into digital documents.

Figure 2. Task Explorer

The next set of questions focused on the parameters of the current method of prototyping tasks that contributed to task disconnects. People complained that transferring or sharing information required many intermediate steps that broke the overall prototyping task. But the harder contention for the task disconnect is the problem of switching between the physical and digital worlds because of the pen-and-paper use. They pointed out that pen-and-paper paradigm also restricted rapid reproduction, edition, undoing and other manipulations of prototype artifacts. Moreover, people claimed they used different media such as images, papers, text, etc. in their prototyping tasks and that their interaction was disconnected because of the need to use devices such as USB drives and CDs to transfer this media from their laptops to desktops and vice-versa. One technology that we think can bridge the disconnect due to the digitization aspects of the pen-and-paper paradigm is the tablet. This is because of the elimination of the need to use pen-and-paper but still have the flexibility to use the pen on the tablet to draw free-form drawings.

DESIGN IMPLICATIONS

From the survey results, we can conclude that the desktop and the laptop are the primary computing devices that people use today to accomplish their work. We also discovered that pen-and-paper is an extremely important device in the area of software prototyping. Because of these collected facts and because the only means which we have for prototyping a solution for task disconnect is software, we decided that the tablet and the desktop computer platforms are a good match for exploring our research questions.

PROTOTYPE DESCRIPTION

The goal in the design of the prototype was to build a user interface that would encompass the values of knowledge continuity and task continuity needed to provide a seamless MPUI user experience.

To understand whether or not we could seamlessly migrate a task between two devices, we needed a means for actually identifying a task. Therefore we created an environment around the task, and called it the Task Explorer [Figure 2].

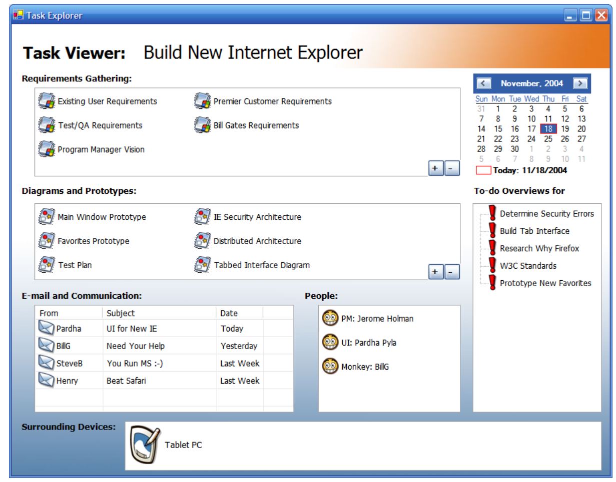

Figure 3. Task Viewer

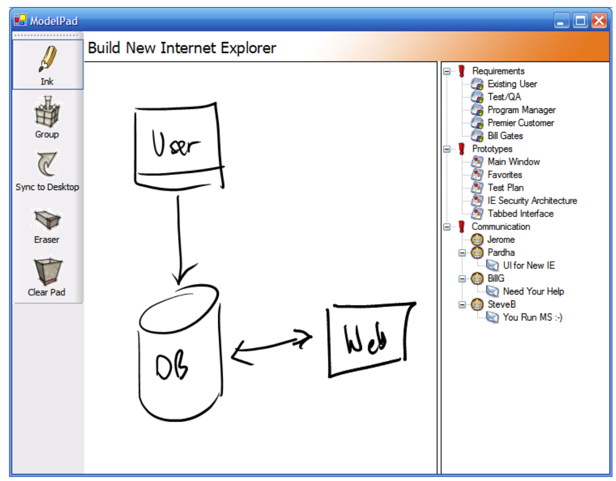

Figure 4. Modeling Tool

The Task Explorer allows the user to create a task, track the activities he or she has to do in an included to-do list tool, and provides constant visual feedback on the status of the connected devices in range.

Opening a task in the Task Explorer launches the Task Viewer [Figure 3]. Within the Task Viewer, the user can see all the related documents and files that correspond to a task. In our prototype application domain, the Task Viewer shows requirements documents, diagrams and prototypes, e-mail addresses, and people related to the project in a unified view. Each task is uniquely color-coded to establish a visual identity with the task. Opening a document such as a requirements specification launches that file in an editor, Microsoft Word in this example, where the user can edit and save changes to the document. Opening a prototype diagram launches our custom modeling tool [Figure 4]. Using this tool, a user can draw and create prototypes and diagrams related to the development task.

The tablet interface leverages spatial organization, shape, color, partitioning of data and function, recovery of state of data and recovery of activity context on its user interface. For every task migrated to the tablet either automatically by the application (because the task was open) or through manual dragging and dropping from the desktop, a full screen window is shown on the tablet with the name of the task and the same unique color gradient per task used to uniquely identify the task on the desktop. The drawing area is automatically loaded with the last drawing that was being accessed on the desktop computer. This is done to automatically recover the state of the data on the desktop, helping maintain task continuity. This also recovers activity context because what is seen is what was last worked on.

If the drawing on the interface is cleared and another drawing is created, the new diagram is created and synchronized automatically with the desktop. This removes the need for opening and saving a document, making the tablet more like paper. As artifacts are being generated, they are populated into the task tree on the right side of the screen. This task tree also brings together the requirements documents, people, to-do list, and email messages to the tablet that were related to the task on the desktop computer.

Using shape and iconic continuity with similar graphics and layout, (but with shortened titles and a more logical data organization), the application leverages the concepts of partitioning of data and function to take into account the rather limited space on a tablet.

Our key objective with this CUI prototype application was to create an environment that promoted knowledge continuity and task continuity in an attempt to bridge task disconnects. By recovering state and activity, and providing an environment that retains the information related to the task at hand, establishing activity context and partitioning that data and functionality in a way that promotes task migration, we feel that we have achieved a first step toward creating an interface that bridges task disconnects.

EVALUATION

The prototype was evaluated with a group of graduate students with a software engineering background. A total of six subjects participated in the study. Three of the six subjects constituted a control group where they were given tasks that required switching between a tablet and a desktop computer. The other three subjects comprised our test group and were asked to perform the same tasks using our prototype. Each participant was given a total of seven tasks. Each task required drawing simple low-fidelity user interface prototypes using our custom drawing tool and updating requirement specifications using a text editor, or a combination of these two activities. The subjects were provided with a background scenario to provide them with the context of a software development project for a fictitious client and the need to transfer documents between tablet and the desktop. The subjects were asked to use a tablet to “meet with the client” and their interaction with the client was scripted in the scenario provided. The participants were asked to think aloud while they were working and the evaluator prompted the users when they stopped talking during a task.

Tasks

The first task required the participant to make changes to an existing requirements document based on the fictitious client’s new insights into the project at the client’s location (i.e. using a tablet). The second task required the subject to prepare a low-fidelity prototype for the new requirements specification on the desktop. The third task asked the client to ‘visit the client’ to demo the prototype that was created on the desktop at the subject’s ‘office’. The fourth task required the subject to work on the desktop and to add more description to some requirements based on the client’s feedback. The subjects were asked to imagine being at home for the fifth task (meaning they were to use a tablet) when they thought of a design feature. They were to quickly create a new prototype with that insight to demo to the client the next day. The sixth task asked the subject to ‘visit the client’ and demo the new prototype and get feedback. The feedback required changing the prototype and the requirement specification. The last task was set to take place at the subject’s office where they were asked to update their desktop files with the latest prototype and requirements specifications.

These tasks were designed with the obvious goal of making the subjects transfer information between the two devices as they progress through the tasks. In the test group, this transfer was automatic because the subjects used our prototype. In the control group, the subjects had to move the files themselves using their choice of a USB pen drive, email, or other server-based technologies. The control group participants were provided with the tablet and the desktop that were both connected to the Internet. They were given one task at a time with the associated scenario to provide the context of the interaction. At the end of the session, all the subjects were asked to fill out a subjective questionnaire.

OBSERVATIONS, INSIGHTS AND CONCLUSIONS

Control Group

For the second task, where the subjects were required to create prototypes based on the requirements specification document, all the three subjects in the control group preferred using the tablet as an information display. They opened the specification document on the tablet and referred to it as they sketched the prototype on the desktop. When asked about this, they said that having the information on a secondary display was good as it did not make them switch between different windows on one platform. This might mean that CUIs should leverage the capabilities of the various devices even when they are co-located. Also, migrating all the information, data and functions into a single device might not be the best way to bridge a task disconnect. We were surprised by this behavior and we believe that this parameter of using devices as additional displays requires further investigation.

For task three in the control group, one of the subjects forgot to copy the files from the desktop to the tablet before “visiting the client”. When provided with the description for task four where the subject realized she had forgotten to get the updated files, she remarked “Wow! In real life this would mean I’d have to go back to my office to get my updated files or redo the prototype that I did in the last task!”

During the course of the evaluation, another user commented “I go through this hassle everyday with my laptop and my desktop. I am always moving files to keep my information up-to-date! It is so frustrating.” The third participant commented “This is very annoying. But it [something] has to be done [because there is no other way]” (words in brackets ours). We claim that this validates our hypothesis that there is a task disconnect due to the use of multiple devices to get everyday tasks done.

One common complaint from the participants was that they had to remember file locations and the state of the file on each platform. As one subject put it, “this version control is getting irritating”. Remembering such extraneous information increases the short term memory costs for this activity tremendously. This, combined with the fact that short term memory of humans is very leaky, one can draw an inference that if the temporal ruptures for a task take place over a long period of time, it is almost impossible for the user to remember which device has the latest version of the data. This is another observation that directly supports our hypothesis that transferring activity context is important.

Another interesting observation that one subject made was: “this [migrating data] almost makes me use the tablet alone for all the tasks and forget about my desktop if I had the choice”. When asked if she would do that even if the task at hand required more processing power (such as that available in a desktop), she responded affirmatively. This hints that task disconnects almost force the users to use a single device alone that is mobile and completely keep away from other devices even if those devices are more suitable for a particular task.

In the subjective questionnaire, all three control group subjects answered that they had a constant fear of making errors due to the overheads associated with migrating data and information across the devices. They also felt it was not easy to keep track of version information for the documents. One participant commented that the real world scenarios such as the ones used in the evaluation session would be worse because of the bigger temporal ruptures for tasks in everyday life (in the evaluation the subjects were performing the tasks immediately after one another).

Test Group

For the second group of three participants, we gave them our MPUI environment and tool to perform the tasks described above. After they familiarized themselves with the environment, we asked them to accomplish the same tasks on both devices, the desktop and the tablet with our application. We observed that they were able to instantly find where the document was and get the information they needed or update and modify it in the way they needed to accomplish the task. When switching from device to device, we found that having the environment loaded with the information automatically allowed them to immediately restart their task and be productive. Because information was redisplayed using less screen real estate, users were immediately able to focus on their work while keeping related information in their peripheral vision. The only limitation of the system was that users spent time moving and resizing the requirements window to enable them to easily see both and work between them.

After they were done with the experiment tasks, we asked them to answer the same questionnaire as administered to the control group. Almost unanimously, our participants felt less likely to make errors accomplishing the tasks. Also, because file state and application state were transferred automatically, the only thing that the users had to worry about was finding the appropriate location in the UI to begin work again. There were comments by some users that it would be nice to have a better view of all the files related to a project, but creating a new file system view was not the purpose of our prototype. Overall, participants of the test group responded that the application was satisfying, interesting, stimulating, and easy to use with the highest ratings on the Likert scale. They also responded (and we observed) that little to no time was spent in transferring files and loading applications, thus allowing all of the users to finish the tasks more quickly and with higher quality.

FUTURE WORK

As described previously in the Control Group subsection, we were surprised by the way users seem to use the secondary platforms as information displays. This aspect of using the available displays to help minimize the switching of views is not currently accommodated in our theoretical grounding. Once we entertain this requirement, a new scenario emerges where there are multiple platforms available in a given context. Therefore, there arises a need to assess the best combination of platforms to constitute the CUI in order to distribute the task in an optimal way. We intend to define more parameters to address these issues. E.g., we envision each platform to have a parameter called *weight*. This weight could be a multi-variable set taking into account the suitability and capabilities of the devices in the CUI. Similarly, the costs associated with the tasks, activities and units should be somehow mapped to the platform in use. This is because the same activity on a different platform incurs different costs (e.g. reading a document on a desktop has different costs compared to reading it on a PDA).

CONCLUSION

We explored the question of how we could enable constructing a seamless transition for a user attempting to complete a task with more than one device, bridging the task disconnect that occurs during the transition. We accomplished this by more specifically isolating our scope to identifying task performance while using two platforms, a desktop computer and a tablet, for a specific application domain of requirements gathering and prototyping. We constructed a prototype that adheres to the principles of knowledge continuity and task continuity in an attempt to create a seamless software bridge over task disconnects. To understand its effectiveness, we subjectively measured user performance while accomplishing a set of requirements gathering and prototyping tasks with software engineering professionals and students while using our prototype and compared the results to that of the same users accomplishing the same tasks using traditional application tools like word processors. Even though we do not claim statistical significance, our evaluation showed that our approach of bridging task disconnects is a promising step in resolving the contention arising due to the everyday use of the multitude of devices that surround us.

REFERENCES

1. R. Bandelloni and F. Paterno. Flexible interface migration. In *Proc. 9th International conference on intelligent user interface (IUI)*, pages 148–155, 2004.

2. V. Bellotti and S. Bly. Walking away from the desktop computer: distributed collaboration and mobility in a product design team. In *Proceedings of the 1996 ACM conference on Computer Supported Collaborative Work*, 1996.

3. A. Bennett. Interruptions and the interpretation of conversation. *Discourse Processes*, 4(2):171–188, 1981.

4. 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.

5. R. G. Capra and M. Pérez-Quiñones. Mobile re-finding of web information using a voice interface. *ACM: Computing Research Repository (CoRR)*, cs.HC/0402001, 2004.

6. R. G. Capra, M. Pérez-Quiñones, and N. Ramakrishnan. Webcontext: Remote access to shared context. In *Proc. Perceptual User Interfaces Workshop (PUI 2001)*, 2001.

7. 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.

8. S. Card, A. Newell, and T. Moran. *The psychology of human-computer interaction.* L. Erlbaum Associates, 1983.

9. 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.

10. C. Chhatpar and M. Pérez-Quiñones. Dialogue mobility across devices. In *Proc. ACM Southeast Conference (ACMSE)*, 2003.

11. 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.

12. 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.

13. K. Drummond. A backward glance at interruptions. *Western Journal of Speech Communication*, 53(2):150–166, 1989.

14. 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.

15. B. John. *HCI models, theories, and frameworks: Toward a multidisciplinary science.*, chapter Information processing and skilled behavior., pages 55–102. Morgan Kaufmann Publishers, San Francisco, 2003.

16. A. Leontiev. *Le Developpement du Psychisme*. Editions Sociales, Paris, France, 1972.

17. J. Leplat. *Tasks, Errors and Mental Models*, chapter Task complexity in work situations, pages 105–115. Philadelphia, 1988.

18. L. W. Lin and T. Kim. Understanding the micronote lifecycle: Improving mobile support for informal note taking. In *Conference on human factors in computing systems*, volume 6, pages 687–694, 2004.

19. D. C. McFarlane. *Interruption of people in human-computer interaction*. PhD thesis, The School of Engineering and Applied Science, The George Washington University, Washington, D.C., 1998.

20. D. C. McFarlane. Comparison of four primary methods for co-ordinating the interruption of people in human-computer interaction design. *Human-Computer Interaction*, 17(1):63–139, 2002.

21. 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.

22. G. Mori, F. Paterno, and C. Santoro. Tool support for designing nomadic applications. In *Proc. 8th international conference on intelligent user interfaces (IUI)*, pages 141–148, 2003.

23. G. Olson and J. Olson. *Groupware and computer-supported cooperative work.*, pages 583–595. The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications. Lawrence Erlbaum Associates, Inc., 2003.

24. A. Parush. The measurement and comparison of cross-device mental models. http://www.carleton.ca/-hotlab/hottopics/Articles/OnMentalModels.html, 2004.