Designing for Seamless Task Migration in MPUIs: Bridging Task-Disconnects

Pardha S. Pyla, Jerome Holman, & Manuel A. Pérez-Quiñones
Dept. of Computer Science, Virginia Tech
660 McBryde Hall (0106), Blacksburg, VA 24060
{ppyla, jeholman, perez}@vt.edu

ABSTRACT
Today, the proliferation of mobile computing has changed the work environment forever. As a consequence, users are forced to orchestrate a complex interaction between multiple devices, moving data and information back and forth, to accomplish their tasks. Users trudge out USB key drives, remote desktop software, e-mail and network file storage in an attempt to mitigate this orchestration. We refer to this break from the task at hand as "task-disconnect." 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 among their devices. We present the theory, definition, and discussion of task-disconnect; our approach towards bridging this disconnect; and our prototype application that was built to be used across the desktop computer and the Tablet PC platforms. We then describe our subjective evaluation to measure the effectiveness of the prototype in bridging the task-disconnect. We then conclude with the results and insights gained from our evaluation.

Author Keywords
Task migration, Multi-Platform User Interfaces, task-disconnect, task continuity, knowledge continuity.

INTRODUCTION AND MOTIVATION
Today, with the deployment of computing in various forms and factors, we work in contention. The face of computing has changed from that of the personal computer world of the 1980s and 1990s. The proliferation of notebook computers and other mobile computing devices continues to change our work environment, driven by the convenience of portable computing. 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 notebook 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. Amy uses her notebook in the library to collect images, references, and to take notes from a few journals in her research area. As she prepares some spreadsheets and graphs to support her points, she sees a call for papers deadline in one of the journals coming up in a few days. She opens the Outlook's calendar program on her notebook and makes an entry in it to remind her the next day to submit an abstract for that journal. After finishing her work at the library, Amy returns to the office to her desktop computer to finish the presentation. She connects a USB key drive to her laptop computer to move the files that she collected and created at the library to her desktop; to cut and paste into her presentation. As she creates the presentation she remembers a paper she read a few weeks ago that could provide some background information for her topic. She tries to remember where she saved that paper. She starts looking in her files on the desktop and realizing it is not there, she looks for it on the notebook computer. 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 notebook is now "out of date", she uses her USB drive again to update that file on her notebook. She leaves for home happy that she is done with the presentation, completely forgetting about the call for papers calendar event on her notebook.

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 point in which Amy is 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 task-disconnect.

BACKGROUND AND THEORY

Before formally defining task-disconnect, we first describe tasks and the various parameters of tasks. A task can be defined as "a goal to be attained in given conditions" [1]. These conditions can be expressed using three points of view: "the states to be covered, the permitted operations, and the procedure"[2]. 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" [2]. We also make use of Leplat's [2] 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. Hence in our work, tasks are composed of activities and activities are further subdivided into units.

For nontrivial tasks (tasks with multiple units), we define 'procedure' to be an operation execution sequence of multiple units. We also associate a parameter required for the successful execution of a unit: instruction. Instructions are knowledge directions necessary to execute units. This knowledge can exist in the user's knowledge of the world or it can exist in the aids and artifacts in the task environment. Another parameter that is fundamental to each unit is cost. Cost is a multidimensional attribute set that is incurred during the execution of a unit [2]. 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. In other words a cost intensive, multi-activity task for one user might be a low cost, single activity task for another depending on the instructions available to that user. It should be remembered that instructions in this context could be tacit knowledge or job aids available to 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.

Task-disconnect

In various scenarios such as Amy's attempt to work with two platforms, the need to transfer task and 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 to literally orchestrate a migration of data back and forth between the two devices create a gap in the task execution. At a high level, we define this gap between the devices and task as task-disconnect. Theoretically, we define 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. That is, 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 disconnection occurs because moving a task from one primary device to a secondary device requires stopping work, opening and loading an assortment of applications on the secondary device to complement or replace the applications being used on the primary device, transferring current data and files to the secondary device, and then opening the information and data with the secondary device's loaded applications to restart work on the original task.

RESEARCH QUESTIONS AND APPROACH

So what does it mean for a task *not* to be disconnected? In other words, how can we maintain task continuity across multiple devices? Task continuity requires the recovery of state and activity context. Recovery of activity context deals with the ability to recover the last actions that were taking place on one device so that they can be taken into account on the other. What if Amy's spreadsheet program transferred the updated copy of her spreadsheet automatically from the desktop computer where she finalized the changes to her notebook when she closed the program? What if the calendar program automatically transferred Amy's reminder to the desktop when she entered her office?

Our goal in this project is to address these questions and to understand how we can have a seamless transfer of information and task across multiple devices to prevent task-disconnects. We define a seamless transition of task between multiple devices to occur if there exist no additional costs; incurred due to activities dealing with temporal task ruptures outside the total costs associated with all the activities required to complete the task at hand.

We characterize such seamless migration to be dependent on knowledge continuity and task continuity [3]. 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 change 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, has an effect on the usability of that program. Poor usability implies that 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 day and see the information for that day is more appropriate. This is an example of data partitioning.

With the theory and background described above, we state our research question: How can we construct 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 that specifically accounts for knowledge and task continuity to seamlessly bridge task-disconnect and subjectively measure the perceived efficiency between using the prototype and traditional disconnected applications when attempting to accomplish a task across multiple devices. We describe the process, prototype, and evaluation in later sections of this paper.

RELATED WORK

Our work has a strong parallel to the traditional Computer Supported Cooperative Work (CSCW) discipline [4]. 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 [5]. They describe Roam, a 'seamless application framework' that can help developers build 'resource-aware' applications capable of adapting to the constraints of various platforms at runtime. However, this 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 [6]. 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 [7]. They then describe an architecture for enabling "dialogue mobility" in applications. They do not 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 [8]. 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. Similarly, Johanson et al. describe a multibrowsing framework using which users can share the visualization of content on their web browsers across multiple displays in an ad hoc computing environment [9].

Biehl and Bailey introduce ARIS, a similar window management framework to "relocate" running applications from one display to another [10]. On a slightly different note, Mori et al. describe a tool called TERESA that helps in designing and developing model-based 'nomadic' applications [11]. They claim that a lack of such automatic tool support is the main reason for the limited deployment of such 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 the users wrestle with while trying to interact with an MPUI.

Denis and Karsenty provide a conceptual framework for "inter-usability" of multiple devices [3]. 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 on this requirement for seamlessness. We take this task centered approach to solving this 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 domains 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 of this application domain is that we have immediate access to 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. One of our team members traveled to Microsoft Campus to talk to software developers and conduct informal interviews. We interviewed a total of six people with experience in software development and prototyping tasks. The process of interviewing was made informal and was more of a discussion between volunteers and our team members. 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.

We also developed an online questionnaire and hosted it on http://survey.vt.edu. We distributed the survey link to our target audience of software developers, graduate students with software development experience, and researchers in HCI who are familiar with computing and do prototyping tasks. We had a total of 32 responses to our survey. We analyzed these responses and the results of that analysis 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 (27 out of the 32 surveyed said they use desktops for their work related activities). Notebooks were the most popular complementary devices other than the desktop (22 out of the 32 surveyed used notebooks for work). This leads us to infer that the computing paradigm today is still geared towards power, mobility and a combination of the two.

Our next group of questions addressed the comfort level, proficiency, and the common practices the users had in a sample application of software prototyping. We asked specific questions regarding the number of prototypes the participants built per project on average, the devices they used to prototype, the task sequence they employed for prototyping (e.g. how often they interacted with clients, how they transferred the artifacts generated in this task between multiple places and stakeholders, etc.), and what factors contributed to any task-disconnects while in this task sequence.

The two devices that are used most for prototyping tasks seem to be the 'pen and paper' and a notebook (27 out of 32 surveyed said they used pen and paper, 11 out of 32 said they used notebooks). The obvious inference here is that people use multiple devices such as 'pen and paper' and notebooks 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 notebooks and 17 said they used desktops to share. We infer that a big group of users actually digitize their sketches as they are (scan paper documents) or into hi-fi prototypes (use drawing tools such as MS Visio) if they have to transfer and share as they selected desktop and notebooks as the devices for sharing. Another question that supported this inference is about the common software technology used to share the prototype artifacts. A majority of 24 people answered email and 11 people used network file sharing, both of which require converting paper and pen artifacts into digital documents.

The next set of questions we asked 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 paper and pen 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 CDROMS to transfer this media from their laptops to desktops and vice a versa. One technology that we think can bridge the disconnect due to the digitization aspects of the pen and paper paradigm is the Tablet PC. This is because of the elimination of the need to use paper and pen but still have the flexibility to use the pen on the Tablet to have 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 that we are investigating. Because of these collected facts and because the only means which we have for prototyping a solution for task-disconnect is software, the Tablet PC and the desktop computer are the perfect platforms for exploring our research question. The Tablet PC which is both a notebook computer and an electronic pen based system, enables us to not only explore how to seamlessly bridge task-disconnect between a notebook and a desktop computer, but it also allows us to explore task-disconnect between the pen and paper and a desktop computer by digitizing the pen and paper with the Tablet PC. The most outstanding design implication that we can take from our survey results is that the network is an acceptable means of transferring information back and forth between devices. E-mail and network storage were popular means of transferring information and files back and forth between multiple devices, and we feel that this justifies using the network as a primary medium for information migration.

PROTOTYPE DESCRIPTION

We constructed an MPUI application prototype for the desktop and Tablet PC devices. We developed the prototype using the Microsoft .NET framework. 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. By doing so, we wanted to establish the use of an MPUI for task migration, bridging task-disconnect between the desktop and the Tablet PC.

The desktop component of the application consists mainly of what we named the Task Explorer. The Task Explorer is the central interface created to help the user establish his or her tasks and related information. To understand whether or not we could seamlessly migrate task between two devices, we needed a means for actually identifying task. Therefore we created an environment around task, and called it the Task Explorer. The Task Explorer, shown in figure 1, 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 a constant visual feedback on the status of the connected devices in range.

Opening a task in the Task Explorer launches the Task Viewer (shown in figure 2). 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 requirement 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 like a requirements specification launches that file in Microsoft Word where the user can edit and save changes to the document. Clicking the plus button adds a new Microsoft Word document with a name selected by the user. Opening a prototype diagram launches our custom developed modeling tool (shown in figure 3). Using this tool, a user can draw and create prototypes and diagrams related to the development task.

The key feature in our prototype is the seamless transition from the desktop user experience to the Tablet PC. As stated above, the Task Explorer and the Task Viewer show any devices that are in range of the desktop computer. You can see in both screenshots of our prototype windows that the Tablet PC is connected to the desktop. When the user takes the Tablet PC and walks out of range of the desktop computer, the Tablet PC opens the Tablet prototyping tool automatically with a view of any tasks that were opened on the desktop computer. The user can also push tasks to the tablet by dragging and dropping their tasks from the Task Explorer to the Tablet PC. In doing this, the user "migrates" their task to the Tablet PC in a seamless manner.

The Tablet PC 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 PC with the name of the task and the same unique color gradient per task used to uniquely identify the task on the desktop. By having the name and gradient color in the same location and of similar size, we use shape, color, and spatial organization as much as possible to ease the migration of task to the Tablet PC.

The Tablet PC's Task window provides a similar toolbar that is available on the desktop PC. This again leverages spatial organization and shape to provide continuity and to make the transition between the desktop and the Tablet PC seamless. The drawing area is automatically loaded with the last drawing that was being assessed 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

Figure 1: Task Explorer

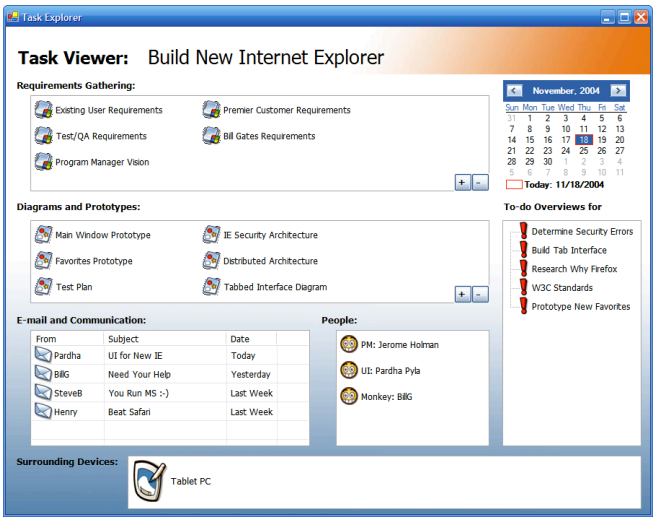

Figure 2: Task Viewer

what was last worked on.

If the drawing on the interface is cleared and another drawing is created, the new diagram is created and synched automatically with the desktop. This removes the need for opening and saving a document, making the Tablet PC 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 e-mails to the Tablet PC 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. This is done by removing the word requirements from requirement documents' names, and organizing e-mails under people's names instead of providing both lists independently. Opening a document from the task tree opens the document in a simplified window that again takes advantage of partitioning of functions to provide a simpler interface for the Tablet PC. For example, the text editor on the Tablet PC is scaled down from the 'heavy' Microsoft Word application as shown in figure 4.

This simplification of filing on the Tablet PC somewhat resembles the "removal or hiding" (from the user's perspective) of the file system on the Palm™ based handheld computers and iApps from Apple™. In these systems, the user is shielded from the intricacies and operations of file handling and storage. The user only has a few file classification capabilities and the data is mostly provided as a service when required. We believe this simplification is one of the important reasons for the success of these platforms.

Our key objective with this prototype application was to create an environment that promoted knowledge continuity and task continuity in an attempt to bridge task-disconnect.

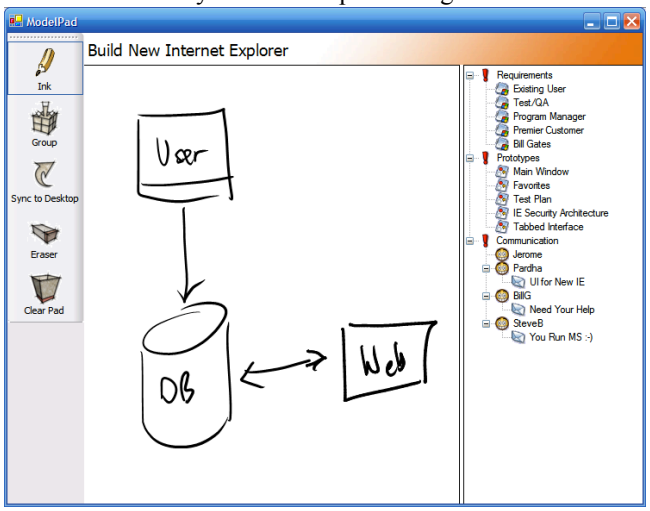

Figure 3: Custom prototyping tool

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-disconnect.

EVALUATION

The prototype was evaluated with a group of graduate students with software engineering background. We used a total of six subjects for the study. Three of the six subjects were used as a control group where they were given tasks that required switching between a Tablet PC 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 made drawing tool; 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 PC and the desktop. The subjects were asked to use a Tablet PC 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 are working and the evaluator prompted the users when they stopped talking during a task. All participants were provided training in using a Tablet PC, the prototype drawing tool, and were given the background about our research questions. The test group participants were provided training with our prototype tool before starting the evaluation session.

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

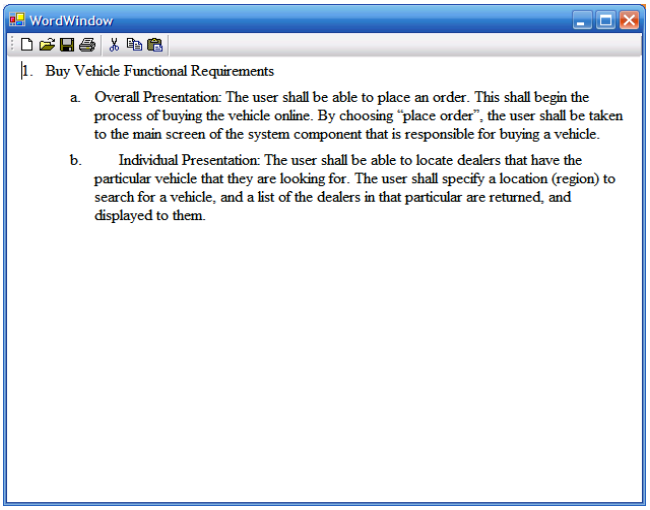

Figure 4: Scaled down text editor for the Tablet PC

(i.e. using a Tablet PC). 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 "client's feedback". The subjects were asked to imagine being at home for the fifth task (meaning they were to use a Tablet PC) when they think 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 PC and the desktop that were 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 task two, 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 PC as an information display. They opened the specification document in the Tablet and were referring to it as they drew the prototype on the desktop. When asked about this they said having the information on a secondary display is good as it did not make them switch between different windows on one platform. This might mean that MPUIs 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 with 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 PC 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 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 that) has to be done (because there is no other way)" [words in parenthesis ours]. We claim this validates our hypothesis that there is a task-disconnect due to the use of multiple devices to get everyday tasks done. We also believe that using approaches like ours in designing applications for multiple devices will change the practice of doing something "annoying" because there was no other way.

One common complaint from the participants was that they have to remember file locations and the state of the file in 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. In such situations there is a need for external instructions (described in the Background and Theory section) for successful completion of a task (e.g. the last modified date on each of the platforms, etc.). 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 requires more processing power (such as available in a desktop), she responded yes. 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 an informal discussion of the findings from this evaluation with our colleagues, we found out that one of the professors in the CS department has actually given up on using a desktop and a notebook combination because of this very reason and only works with a notebook computer.

In the subjective questionnaire, all three control group subjects answered 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 on another). On the question about the percentage of time spent on activities

that were not directly related to the task at hand (i.e. the overhead in getting the tasks done), all three subjects answered that about 30% was spent on task overheads. We believe that to be a reasonable and representative amount of overhead in a complex task such as software requirements gathering and prototyping.

Test Group

For the second group of three participants, we gave them our MPUI environment and tool to perform the tasks described above. We introduced the group to the tool, allowing them to explore the interface and understand the limitations and the capabilities of our prototype application. After they were familiarized with the environment, we asked them to accomplish the same tasks on both devices, the desktop and the Tablet PC with our application. We then observed, asking them to speak out loud as they used the tool, and we recorded our observations.

One thing that was definitely apparent is that having an environment built upon task enabled them to easily understand where they needed to go to get information. 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 the tool on the Tablet PC was loaded from the start, there was no time delay in switching between devices. The user was immediately able to find their place and keep moving. Because information was redisplayed using less screen real estate, users were immediately able to focus on their work while keeping their 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. One immediate observation is the decrease in the fear of making errors. Almost unanimously, our participants felt less likely to have 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. As for the other questions, users found it easier to track their progress and information compared with the control group. There were comments by 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 used in transferring files and loading applications allowing all of the users to finish the tasks more quickly and with higher quality because they could focus their energy on the task at hand and not the overheads.

In conclusion, we explored the question of how we can construct 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 PC, 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-disconnect. 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 Microsoft Word. Even though we do not claim statistical significance, our evaluation showed that our approach of bridging the task-disconnect is a promising step in resolving the contention arising due to the everyday use of the multitude of devices that surround us.

REFERENCES

1. Leontiev, A. *Le Developpement du Psychisme*. Editions Sociales, Paris, 1972.
2. Leplat, J. Task complexity in work situations. in Goodstein, L.P., Andersen, H.B. and Olsen, S.E. eds. *Tasks, Errors and Mental Models*, Taylor & Francis, Philadelphia, (1988), 105-115.
3. Denis, C. and Karsenty, L. Inter-Usability of Multi-Device Systems - A Conceptual Framework. in Seffah, A. and Javahery, H. eds. *Multiple User Interfaces: Cross-Platform Applications and Context-Aware Interfaces*, John Wiley & Sons, (2004), 373-384.
4. Olson, G.M. and Olson, J.S. Groupware and computer-supported cooperative work. in Jacko, J.A. and Sears, A. eds. *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications*, Lawrence Erlbaum Associates, Inc., (2003), 583-595.
5. Chu, H.-h., Song, H., Wong, C., Kurakake, S. and Katagiri, M. Roam, a seamless application framework. *The Journal of Systems and Software*, *69*, (2004), 209-226.
6. Bandelloni, R. and Paterno, F., Flexible Interface Migration. *Proc. 9th International conference on intelligent user interface (IUI)*, (2004), 148-155.
7. Chhatpar, C. and Pérez-Quiñones, M.A., Dialogue mobility across devices. *Proc. ACM Southeast Conference (ACMSE)*, (2003).
8. Florins, M. and Vanderdonckt, J., Graceful degradation of user interfaces as a design method for multiplatform systems. *Proc. 9th international conference on Intelligent user interface*, (2004), 140-147.

9. Johanson, B., Ponnekanti, S., Sengupta, C. and Fox, A., Multibrowsing: Moving web content across multiple displays. *Proc. 3rd international conference on ubiquitous computing*, (2001), 346-353.
10. Biehl, J.T. and Bailey, B.P., ARIS: An interface for application relocation in an interactive space. *Proc. 2004 conference on graphics interface*, (2004), 107-116.
11. Mori, G., Paterno, F. and Santoro, C., Tool support for designing nomadic applications. *Proc. 8th international conference on intelligent user interfaces (IUI)*, (2003), 141-148.