RelAltTab: Assisting Users in Switching Windows

Nuria Oliver, Mary Czerwinski, Greg Smith & Kristof Roomp
Microsoft Research & Microsoft Corporation
Redmond, WA, USA
nuriao@tid.es,{marycz,gregsmi,kristofr}@microsoft.com

ABSTRACT

We present RelAltTab, an enhanced ALT+TAB prototype that assists users in switching windows. Our approach uses semantic and temporal information to create a list of *related* windows to the window that the user is currently engaged in. The main assumption is that the user is more likely to switch to a *related* window than to any other window in the system. We propose two different user interfaces that present the related window list to the user. We describe in detail the techniques and user interfaces of the RelAltTab system, and present the results of one user study comparing our approach to the standard Windows ALT+TAB program.

Keywords: Intelligent Window Switching Applications, ALT+TAB, Window Similarity Modeling, Related Windows

General Terms: H.5.2 [Information Interfaces and Presentation]: User Interfaces; D.2.2 [Software Engineering]: User Interfaces; I.5.4 [Pattern Recognition]: Applications

INTRODUCTION & PREVIOUS WORK

Application switching has been necessary since computers allowed users to multi-task. With the advent and ubiquity of graphical user interfaces and the desktop metaphor, window switching has become one of the most common operations performed on a computer.

The techniques for window switching have been categorized into three approaches [3]: a. *Temporal*: Windows are presented to the user based on their time of last access; b. *Spatial*: Spatial approaches may use an initial ordering based on temporal information or on where the window is located on the screen. The relative order of the windows in the window switching view does not change, unless there is a change in the number or spatial location of the windows; and c. *Hybrid*: Hybrid approaches use a combination of temporal and spatial techniques to determine how to present the list of windows to the user.

ALT+TAB is the canonical example of a *temporal* approach. It ranks the window icons or thumbnails in Z-order, which is related to the order with which the windows were accessed by the user.

The Windows Taskbar follows a spatial approach in its organization of the window icons. The user has direct access to any open window by clicking on a button or iconic representation of the window. The location of the icons or buttons on the Taskbar are fixed and therefore this approach takes advantage of the user's spatial memory. In addition, the taskbar groups windows within the same application by default onto one button to save space. Another example of a spatial approach is Apple's Exposé, which uses a spatial layout to arrange the windows on the desktop in a visual representation.

Hybrid approaches use temporal ordering, but also allow for random access. The new Windows Vista^{TM} ALT+TAB interface shows live thumbnails of all the desktop windows and allows users to either cycle through them by repeatedly pressing ALT+TAB or to directly switch to any window by clicking on its thumbnail with the mouse.

Finally, the EyeExposé system [3] presents an innovative extension of the Exposé system by allowing users to utilize a combination of keyboard and eye gaze. This approach combines the use of a two-dimensional layout visualization for showing the user all open applications and the use of eye gaze tracking for selecting the desired window.

The system proposed in this paper adds a new criterion to determine the ordering of the windows in the ALT+TAB user interface: the *semantic similarity* of the window titles. In our previous work with the SWISH system [4], we successfully used both semantic and temporal similarity between windows to automatically cluster them into tasks. The RelAltTab system leverages our experience with SWISH, with which it shares several features.

ARCHITECTURE

The RelAltTab prototype is implemented as a Windows executable that registers itself on the target system as being the ALT+TAB handler for the OS, so that it is automatically triggered when the user invokes ALT+TAB. It incorporates a component called the WinModel, based on the VibeLog [2] logging tool to provide the framework for computer activity collection.

WinModel uses the public Windows WinEventHook event stream to build and maintain a live, in-memory window model, representing the current window state of all the top-level windows in the system. RelAltTab processes the window objects supplied by WinModel on every significant change to build a representation of "relatedness" among windows, and to provide the appropriate user interface when ALT+TAB is invoked.

RelAltTab monitors and logs the following events with their corresponding timestamps: foreground window changes with window titles, invocations of ALT+TAB with the position of the ALT+TAB selection and total number of windows on the desktop, and a record

of whether the current foreground window was in the *related* list of windows, according to our model.

WINDOW SIMILARITY MODELING

The data analysis in the RelAltTab prototype is derived from the assumption that windows that are related to each other –based on a particular criterion– share some features. This assumption is motivated by the observation that tasks typically involve multiple windows and those windows have something in common. The next assumption is that, during the course of a working day, the user is *more likely* to switch to windows that are related to each other (and hence belong to the same task) than to windows that are not.

As we did in the SWISH system [4], we focus in this paper on two sets of features:

1. **Semantic features**: Windows that are related to each other share common words in their content and in their window titles. In order to quantitatively measure the semantic similarity of two window titles, (t_i and t_j), we carry out the dot product of their *tfidf* vectors, $dot(tfidf_i, tfidf_j)$ [4]. This dot product gives the correlation between the terms over the titles (documents). In our analysis, the higher the dot product, the *more semantically similar* the titles are.

At each instant of time, the RelAltTab prototype computes the dot product between the foreground window's *tfidf* vector and the *tfidf* vectors of all the other N windows in the system. The result is an N dimensional *semantic vector*, v_{sem}. This vector will be used to create the list of *related windows*.

2. **Temporal features**: Windows that are related to each other are accessed in temporal proximity to each other, *i.e.*, input focus switches between windows belonging to the same task occur more frequently than switches to windows outside the task. RelAltTab keeps track of all the window switching events that have taken place during a time interval of length T, and automatically builds a *window switching* matrix, WS, where each element ws_{ij} is proportional to the number of times that the user switched from window w_i to window w_j during the time period T. In our experiments, we used a time window T of length 10 minutes.

If the foreground window has index k in WS, a *temporal vector* (v_{tem}) is created by sorting the kth row of WS. In our analysis, the higher the values in this vector, the *more temporally similar* the windows are.

The RelAltTab prototype builds a list of *related windows* to which the user is more likely to switch. This list of related windows is created by combining the semantic and temporal vectors. A window i in the system is included in the list of *related windows* if: (1) $v_{sem}(i) > t$, with $t = 1$; or (2) $v_{tem}(i) > 0$. Note how a temporally related window will change its status if the user stops switching to/from that window for a period of time. Also note that a window could be both semantically and temporally related.

RELALTTAB USER INTERFACES

In order to evaluate the RelAltTab system, we have developed two user interfaces that we will refer to as Prototype B and Prototype C. We have also developed a *baseline* system, named Prototype A, that has the same functionality as the standard Windows ALT+TAB and similar UI.

Prototype A

Prototype A is our baseline system. It offers the same functionality and similar looks to the standard Windows ALT+TAB program. The user interface is depicted in Figure 1 (a). As it is shown in the Figure, the prototype shows the thumbnails of all the windows in the system. If a window is minimized, its icon is displayed. In addition, the window title appears underneath the thumbnail. The current position of the ALT+TAB cursor is highlighted with an orange frame around the thumbnail, and the title of the window is shown on the top of the interface. As with the standard ALT+TAB program, the user can press ALT+TAB to move the cursor forward or SHIFT+ALT+TAB to move it backward.

The ordering of the thumbnails in Prototype A is the same as in the ALT+TAB program: (1) *Maximized and Top-most windows* appear at the very beginning of the thumbnail list; (2) *Minimized windows* appear at the end of the thumbnail list; (3) The rest of the windows appear in Z-order, which is related to the recency of access by the user: windows that were more recently accessed appear closer to the beginning of the thumbnail list than windows that were accessed less recently.

We did not use the built-in ALT+TAB interface as a baseline for two reasons: First, since our user interface was somewhat different than the built-in UI, we wanted to avoid introducing artifacts that could affect our comparative analysis. Secondly, the built-in ALT+TAB interface does not allow attaching the monitoring and logging functionality that we needed in order to collect baseline statistics. We shall highlight, though, that we designed Prototype A to look and behave as similarly to the shipping ALT+TAB as possible. In our experiments, none of our participants raised any concerns with Prototype A being too different from their standard ALT+TAB.

Prototype B

As shown in Figure 1 (b), the user interface looks similar to that of Prototype A. However, the *ordering* of the thumbnails on the interface is very different from that of Prototype A. While *maximized, top-most and minimized windows* are treated the same way as in Prototype A, the rest of the windows are treated differently: The first 3 thumbnails in the list appear in Z-order and therefore are the same as in Prototype A. Next, the system displays the thumbnails of all the windows that are *semantically* related to the current foreground window and it highlights them with a salmon colored background. The *temporally* related windows appear next and finally the rest of the windows are shown in Z-order.

Prototype C

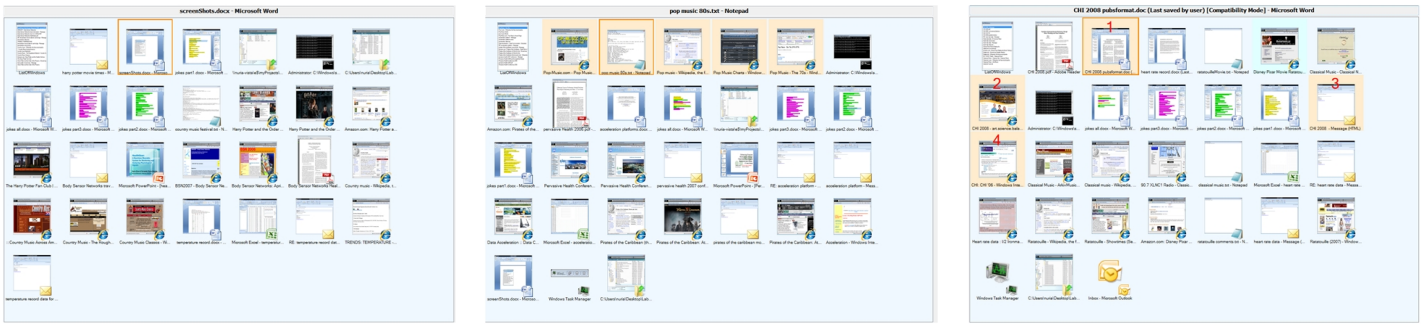
In Prototype C, we wanted to include a *random access* capability while preserving the standard Z-ordering of the thumbnails. Figure 1 (c) illustrates the interface. As seen in the Figure, Prototype C introduces two types of background coloring: salmon for the semantically related windows and light-blue for the temporally related windows. It also displays a red number on top of each semantically related window. This number allows the user to directly switch to the window by pressing ALT+NUMBER. For example, if the user wants to switch to the last window in the second row in Figure 1 (c), he/she only needs to press ALT+3 to directly switch to that window.

COMPARATIVE USER STUDY

In order to compare the three ALT+TAB prototypes, we carried out a user study in our laboratory. We were interested in two measures: (1) task performance as measured by the completion time, and (2) subjective evaluation via post-study questionnaires.

The study consisted of 2 parts:

1. *Random Access:* The first part of the study was a random access task, where the user was asked to find and switch to a series of windows, one at a time. The computer desktop was set up with 25 windows, 20 of which were part of this random access task. The



(a) Prototype A (b) Prototype B (c) Prototype C

Figure 1: User interfaces for Prototypes A, B and C.

windows were grouped in 4 different tasks of 5 windows each. The windows belonged to common applications, including Microsoft Word, Internet Explorer, Notepad, Microsoft Excel and Outlook. The 4 tasks were about *movies*, *music*, *conferences* and *data*. Three isomorphic sets of each of these tasks were devised so that they were equivalent to each other. For example, the *movies* task consisted of Internet Explorer and Microsoft Outlook windows about the Ratatouille, Harry Potter or Pirates of the Caribbean movies.

After reading some instructions on the computer screen, users pressed "OK" and a window appeared on the top-right corner of the monitor with a list of 20 window titles of the 20 windows that they were asked to find and switch to. The list was randomly created according to the following criterion: the order of the tasks was randomly chosen first (*e.g.* movies, data, conference and music). Within each of these tasks, the windows were also randomly selected. All the windows within a task would appear next to each other (in random order), before switching to the first window of the next task. Participants were then asked to find and switch to these 20 windows, one at a time and in the displayed order, only using the ALT+TAB key combination. The current window title that participants were asked to switch to was highlighted in blue.

2. *Copy and Paste:* The second part of the study was a cut-and-paste task. In addition to the windows from the random access task, the computer screen had 3 Microsoft Word documents named *jokesN.docx* (N=1,...,3) with 10 jokes each, and an empty Microsoft Word document named *jokesAll.docx*. During this part of the study, participants had to switch to each of the *jokes* documents in sequential order, read all the jokes in them, select their favorite joke and copy it to the *jokesAll.docx* document. Once they had found their favorite joke on each of the 3 *jokes* documents, participants were asked to repeat the same task but with their *least favorite* joke. Therefore, by the end of this part, the *jokesAll.docx* document would have 6 jokes, 3 favorite and 3 least favorite. To facilitate the task, all the jokes in each of the documents were highlighted with a different color (magenta, green and yellow, respectively). In addition, participants were interrupted one to four times during this task at some random times. When interrupted, a new window would appear (covering the entire screen) and it would ask them to switch to a randomly chosen window in the system. Once they had switched to that window, they pressed "OK" on the interruption window and they were asked to resume the "jokes" task. With each interruption, we were interested in measuring the *random access* and *task resumption* times.

Method

Twenty six participants (9 female and 17 male), with an average of 15.6 years of computer experience and experienced Microsoft Office users as identified by a validated screener were recruited for this study. Participants had a broad set of computer-related jobs, including software design engineers, researchers, administrative assistants, product designers, technical writers and procurement specialists. Most of them (80.8%) spent at least 9 hours working on the computer and reported having 6 or more windows simultaneously open on their computer on a typical day (84.6%). Seven of them (26.9%) reported rarely or never using the ALT+TAB program to switch between windows on their desktop. Interestingly, all of these participants except for one reported having 10 or less windows open on their desktop in a typical day.

Participants arrived in groups of 3 at a time, and each session lasted for about 1.5 hours. Upon arrival, participants were given a brief presentation with instructions about the overall study procedure, the tasks that they were about to perform, and pictures of the three prototypes that they were going to use. The study was run on three identical, late model Compaq Evo machines with a single flat panel LCD monitor running at 1024×768 resolution, each of them running one of the prototypes. Late model Microsoft keyboards and standard mice were used for input. Windows Vista^{TM} with Internet Explorer 7 and Microsoft Office 2007 were the base OS and applications used in the study. The computers already had all the 25 windows that were part of the study open and organized in random order. Participants had to carry out the two above-mentioned tasks with each of the prototypes. The order of trying out the prototypes was counterbalanced across participants to avoid bias.

All three computers logged the following interactions during the study: every foreground change, with its timestamp and window title, a timestamp for the beginning and end of each of the parts of the study, and a record of whether the current foreground window was in the *related* list of windows, according to our model. Dependent measures collected included task time, subjective satisfaction responses to a questionnaire presented after using each prototype, and overall prototype preference. Task times were computed from the logged information.

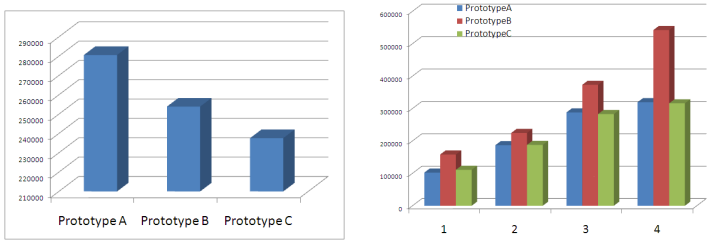
RESULTS

Task Times

First Task: Random Access

A one-way analysis of variance with repeated measures (RM-ANOVA) was performed on the task time data. A significant main effect of user interface was obtained, F(2,50)= 3.3, p=.045. Paired contrasts revealed that the only user interfaces that were significantly different from each other were Prototype A (baseline condition) and Prototype C. The two alternative ALT+TAB user interfaces were not significantly different from each other. The average task times

for the 3 user interfaces can be seen in Figure 2 (a).



(a) (b)

Figure 2: Average task times (in milliseconds) for the (a) first task (*i.e. random access*), and (b) for the second task (*i.e. copy and paste*) with respect to the number of interruptions (horizontal axis).

Second Task: Copy and Paste

The second task did not show any significant difference in task completion times between the baseline condition (Prototype A) and Prototype C. However, participants took significantly longer time with Prototype B, especially as the number of interruptions increased (see Figure 2 (b)). This result makes intuitive sense: this part of the study required switching back and forth between two or three windows. Prototypes A and C behave almost identically in this situation. By contrast, Prototype B reorders the windows that appear in the ALT+TAB interface when the user selects a new foreground window. Therefore, it took participants a longer time to switch back to the "jokes" task after an interruption. Figure 2 supports this intuitive explanation with our study data.

User Satisfaction

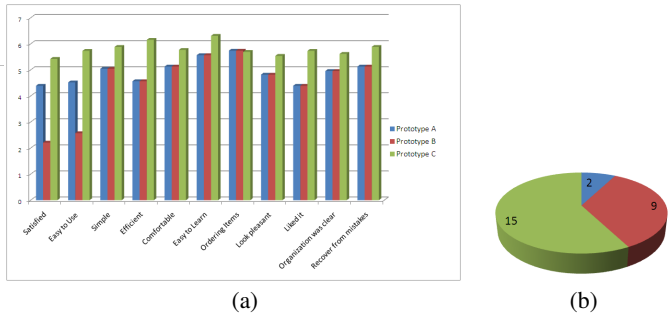
A 3 (user interface) x 11 (questionnaire question) RM-ANOVA was carried out on the user satisfaction ratings. As with the task times, there was a significant main effect of user interface tested, $F(2,48)=4.0$, $p=.025$. In addition, there was a significant effect of questionnaire question, $F(10,240)=7.55$, $p<.001$, and a significant interaction between user interface and questionnaire item, $F(20,480)=2.58$, $p<.001$. Once again, post-hoc analyses revealed that the only two user interfaces that differed from each other significantly were user Prototypes A and C, while B and C were not reliably different in the questionnaire ratings. The interaction can be explained in that some of the questions uncovered user interface differences more strongly than others. The user satisfaction ratings for the 3 user interfaces can be seen in Figure 3. Note how Prototype C was rated higher than any other prototype in *all* the measures except for one, corresponding to the ordering of the items on the interface.

Figure 3 (b) shows how only 2 participants preferred Prototype A (baseline) over the other two prototypes. It is interesting to note that these two participants happened to be very experienced ALT+TAB users who typically had very few windows open on their desktops.

CONCLUSION

We have presented RelAltTab, a system for window switching that uses temporal and semantic information to create a list of *related* windows to the foreground window.

We have developed two user interfaces (Prototype B and Prototype C) that present the *related window* list to the user. We have evaluated our prototypes in a laboratory user study. The results have been encouraging: The *related window* list contained the next window that the user switched to in over 80% of the instances.



(a) (b)

Figure 3: (a) User satisfaction ratings for prototype A, B and C. (b) Preferred user interface: Blue (Prototype A), red (Prototype B) and green (Prototype C).

We have found a significant positive effect of Prototype C when compared with our baseline interface (Prototype A). Participants were *significantly more efficient* in completing a random access task with Prototype C than with Prototype A. In addition, participants showed a *significant preference* for Prototype C over the baseline system. We have also found a negative effect of *reordering* the thumbnails on the user interface when the user frequently interrupts the main task to switch to an unrelated window.

Despite great research efforts in the areas of task modeling, identification and management, there has been fewer innovation in the area of user interfaces for window switching. At the same time, it has been reported that a very small percentage of users [1] regularly use the ALT+TAB key combination for window switching. Perhaps this low penetration reflects a need for innovation and additional functionality in desktop systems for window switching. We believe that our work with RelAltTab points to an opportunity to develop more intelligent window switching systems.

Some areas of future work include: Carrying out a comparative analysis of efficiency of icon-based task switching versus ALT+TAB to find out whether the low penetration of ALT+TAB usage is based on lack of efficiency, lack of user knowledge or habit. Finding out how well the user expectations match what the computer is showing them. Ideally, the task switching system would require as little cognitive load as possible, so that users could concentrate their energy on the actual tasks that they are performing, rather than on switching between them. Carrying out a long term study over a period of months to have a better understanding of the impact of RelAltTab in the user's daily computing experience.

REFERENCES

[1] M. Czerwinski *et al*. Toward characterizing the productivity benefits of very large displays. In *Human-Computer Interaction. INTERACT '03*, 2003.

[2] D. Hutchings, G. Smith, B. Meyers, M. Czerwinski, and G. Robertson. Display space usage and window management operation comparisons between single monitor and multiple monitor users. In *Proc. of the working conference on Advanced Visual Interfaces, AVI 2004*, 2004.

[3] M. Kumar, A. Paepcke, and T. Winograd. Eyeexpos'e: Switching applications with your eyes. Technical report, Stanford University, February 2007.

[4] N. Oliver, G. Smith, C. Thakkar, and A. Surendran. Swish: semantic analysis of window titles and switching history. In *Proc. of the Int. Conf. on Intelligent User Interfaces, IUI 2006*, 2006.