WYSIWYG-Tool Tips: Enhancing Tool Tips with Translucent Preview Bitmaps

Heiko Drewes and Albrecht Schmidt

Embedded Interaction Research Group
Media Informatics, University of Munich, Germany

{albrecht, heiko}@hcilab.org, www.hcilab.org

Abstract. This paper suggests to enhance the concept of tool tips by presenting translucent preview bitmaps. The basic idea is to give the user a preview of what would happen when a certain operation is invoked. We have implemented two prototypes to evaluate the concept. One provides WYSIWYG (What You See Is What You Get) tool tips for documents and applications on the desktop. The second one is an application that provides previews of dialogs while browsing through the menu. Initial experience shows that this can provide a benefit for the user as she or he sees what is going to happen rather than only providing feedback on what has happened. This is in particular of interest to actions that can not be undone.

1 Introduction

A central rule when designing interactive systems is to give the user control over the system [1]. To feel comfortable and in control when interacting with a computer, it is necessary to have an expectation about what will happen as a result of the intended action. The user interface combined with the user's knowledge must allow to predict what a command or action will result in. In graphical user interfaces descriptive names of menu items give the user a hint what will happen when she or he is invoking this menu item. For further assistance of the users, it is common to provide more information about a menu item by displaying a tool tip. Such a tool tip (or balloon help) is a small window, which displays a textual or iconic description of what to expect when choosing the item under the current mouse cursor position.

It is hard to provide useful textual tool tips. On one hand they should be short so that the user can instantly recognize the message. On the other hand if the text in a tool tip is too long it becomes cumbersome to read all the text. Tool tips normally display the menu item text and do not carry more information. It is common to display additional help texts in the status bar of an application, but very often just the name of the menu item is repeated or the menu item text is made into a sentence., e.g. 'Save' and 'Saves the document'.

The following example shows shortcomings of current textual tool tips. The tool tip that commonly appears when the mouse cursor is over the tool bar button for saving a document in Microsoft Word says "Save Document". The resulting action on pressing the button is however different depending on whether the document was

already saved or whether it is the first time that the document is saved. In the first case no dialog will appear and the document will be saved in the background. In the second case a dialog will appear where the user can choose a name for the document. Similarly the resulting action on pressing the print button in the toolbar can not be determined from the label or tool tip in current applications. E.g. in Microsoft Word it commonly prints the whole document to the default printer whereas in the Adobe Acrobat Reader a dialog appears. This motivated the development of prototypes, where additionally to the tool tip a semi transparent bitmap preview of the result of the action is displayed.

For the convenience of the users, most software applications provide several ways to invoke the same dialog – from the menu, from a toolbar, from a property context menu or even from outside the application by the control panel. Multiple ways to the same dialog support the intuitive use. The other side of the coin is, that if a user is searching for a specific setting, she or he may end up again and again in the same dialog, which was searched for the specific setting already. Here too providing a preview of what is going to be displayed when selecting an option can make searching quicker.

2 Preview and Feedback

Feedback is an important topic for human computer interaction. Providing a confirmation that the intended action had taken place is reassuring for the user [1]. With the spread of computers the question in the users head shifted from 'Does the computer do this?' to 'How to get the computer to do this?'. So the answer to the question how to assist the user controlling a computer shifts from feedback to preview. A preview message like 'This will delete all your data' is more helpful to the user, than the feedback message 'All your data have been deleted'.

In cases where interaction with the computer has a permanent effect and where undo is not possible, feedback is too late for guiding the user. Typical tasks in desktop applications that have a permanent effect are printing or sending email. These actions can not be undone as they either change the physical world or have effects in areas where the user has no control anymore. When looking beyond desktop systems into physical computing or embedded computing this becomes even more critical, e.g. shutting down a pump in a power plant or accelerating a vehicle will in general have effects that are not easily reversible.

For communicating what is going to happen we considered different options for presentation, such as texts, icons, and graphics. From informal user interviews we learned that people are reluctant to read long tool tips and that icons require learning of their meaning. To make a preview work it has to be instantaneously recognizable by the user. Therefore the best options are images of what will happen. However presenting solid and large images on top of the application will hide information. This is an annoying effect with the existing small textual tool tips. Also solid images as a preview for dialogs can not be distinguished from the real dialog. The main two parameters of interest for WYSIWYG tool tips are the size and the transparency of the presented preview picture. To explore this further we have implemented prototypes.

3 Application Tool Tips

The basic concept of providing previews of dialogs that will appear is implemented in the application tool tip prototype. When moving the mouse over the toolbar a preview of the dialog, that would appear when this button is pressed, will be shown additionally to the textual tool tip (see figure 1). This is implemented similarly for menu items as menu tips. When a menu item is highlighted a preview of the dialog, which will appear when this action is chosen, is presented, see figure 2.

The application demonstrated is an empty Win32 application, generated by the framework of the integrated development environment, extended with the dialog preview. Based on this application example any application can be developed including WYSIWYG tool tips, but it could also be implemented into the operating system without the need to code it within any application. Whenever the mouse cursor stays a while over a button on the toolbar or an item of the menu, a translucent preview of the corresponding dialog appears at exactly the position it will appear when clicking the item. In our prototype we can change the value for transparency. An extensive discussion on the use of transparency can be found in [2]. Alternatively to the use of transparent visualization multiblending, as introduced in [3], could be used.

The save button of this application shows the typical behavior of applications. It behaves different depending on whether the document has been saved already or not. The first time a document is saved, the application pops up a save dialog to enter a filename. Pressing the save button again does not show a dialog but simply saves the document under the given name. This is reflected in the bitmap tooltip.

Presenting a preview of the dialog that will appear has two advantages. One is speed. The user can already see in the dialog preview if this provides the options she or he needs. In general this saves two clicks (one for opening the dialog and one for canceling the dialog) and the mouse movement in between. The second advantage is to avoid unwanted actions, e.g. when there is no preview for the save action it is clear that the user will not have the chance to give the document a new name.

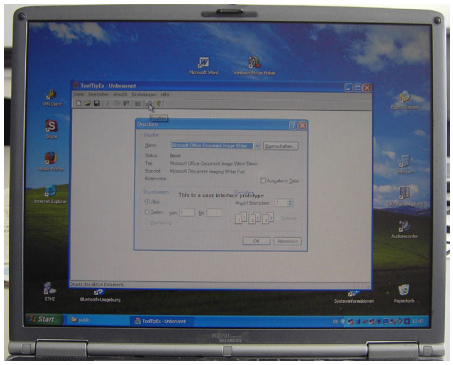

Figure 1: When the mouse cursor is over a button in the toolbar, additionally to a textual tool tips a preview of the dialog to expect is visualized.

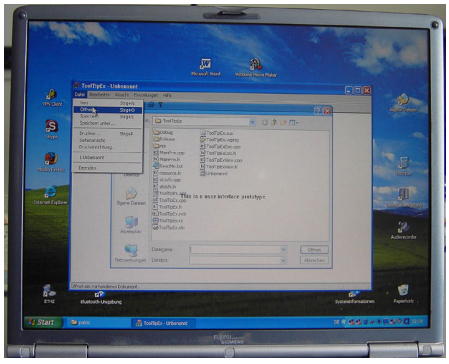

Figure 2: While navigating in a menu a preview of the expected dialog is presented. This allows the user to avoid choosing menu items which lead to unwanted dialogs.

4 Desktop Tool Tips

In the desktop tool tips we generalize the concept to the preview of applications and documents. The basic idea demonstrated in this prototype is to provide a preview of what happens when an icon on the desktop is double clicked (or clicked depending on the setup of the system).

This prototype implements translucent previews of applications for the icons on the desktop. Whenever the mouse cursor stays a while over an icon on the desktop, a translucent preview of the application appears on the desktop. On an application icon the preview shows the opened application, on a document icon the preview shows the opened application with the document loaded.

In the current version of the prototype the previews are static images and taken from a preconfigured bitmap. For a product implementation we would expect that the application will store a bitmap of the window when closing together with the position where the window was presented. This can be done by the operation system using the representation in the display memory.

When moving the mouse pointer over a desktop icon of an application a preview of the application as it will open appears translucent on the screen. It is visualized on the desktop how and where the application would start. In figure 3 this is shown for a document. A translucent picture of the application, as last opened, is presented. In this case it shows that the document was open in full screen and would open in full screen if double clicked. We extended the prototypes for our experiment to allow to select the level of transparency and the size of the window.

5 Implementation of the Prototypes

Both prototypes are written for the Windows XP platform using C++ and Visual Studio. They make use of functionality of the operating system to overlay transparent windows. The prototypes are available for download in the internet on the project web page, see [4].

Both prototypes catch notifications from the operating system. The application tool tips prototype does this by overriding the GetMessageString-function of the framework, which is asking for a help text for the status bar. The desktop tool tips prototype runs in the background. It installs a system wide hook, which catches the notifications from the operating system for displaying and hiding a tool tip (TTN_SHOW, TTN_POP).

In contrast to the application tool tips prototype, where all message handling takes place within the same address space, the desktop tool tips prototype has to handle messages and share data across processes. As a technical consequence, the hook function must reside in a dynamic link library, as dynamic link libraries map their address space into each of its client processes, see [5].

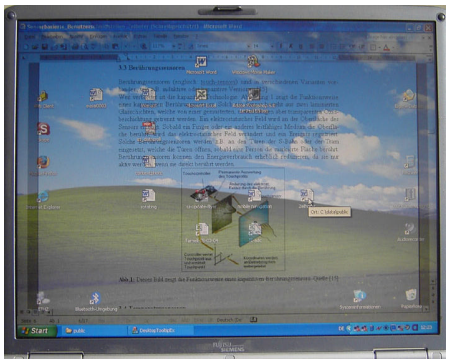

Figure 3: When the cursor is positioned on a document the translucent preview of this document is presented.

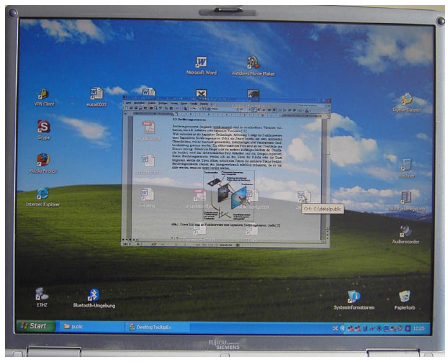

Figure 4: In the interviews users could experiment with the settings of transparency and size of the preview.

6 Interviews with Users

We have used the prototypes to discuss the idea and design considerations for WYSIWYG tool tips. Over the last month we conducted informal interviews with users from different backgrounds.

When presenting the application tool tips prototype, nearly all persons stated something like: “This is nice, but I know how an open or save dialog looks like. The idea will make more sense for dialogs which I do not know.” This statements suggest that it is important to use a real and complex application for a larger user study. In general there was agreement that the presentation of the dialog to expect is providing additional control over the system as one knows what the system will do. Most users were happy with the visualization of the dialogs in real size and did not feel that it obstructs their view on the document, as they are navigating in a menu or a toolbar anyway.

Talking with the users about the desktop tool tips gave similar results. People normally know how an application looks like if they put it on their desktop, but they appreciated the preview of their documents. Some of the users asked for the preview of the documents laying on the desktop toolbar. Others asked for such previews in the explorer for documents like Word, PDF and HTML files. The existing thumbnail previews are helpful, but it is impossible to read a line of text in this reduced bitmap.

With regard to size most users preferred larger previews of documents and applications (50% to 100% of the original size) as this allowed them to really see the document details. In terms of transparency the preferred levels depended on the used background and on the size of the preview. It appears that users would like to have full size previews with a high level of transparency and that smaller previews are preferred with less transparency, see figure 3 for an example of a full screen 60% transparent document preview and figure 4 for a half size 30% transparency preview of the same document.

One of the interview partners gave a hint to a psychological effect. Clicking or double-clicking something invokes an action. If it is not clear what kind of action will take place, a shy user avoids to click anything; maybe this click will destroy something or change some settings and the system won't work anymore. Moving the mouse doesn't invoke any action and the preview of a dialog or an application does not have the obligation to read the displayed text and to decide, whether to press 'OK' or 'Cancel'

7 Conclusions

In this paper we introduced the concept of translucent WYSIWYG tool tips to provide preview information on action that may happen if the user invokes a command. In two prototypical implementations we demonstrated how this can be realized within applications and on the desktop.

Providing a preview helps the user to anticipate what will happen if a button is pressed. From the participatory design sessions it can be concluded that such functionality is helpful when working with a new, unknown (e.g. guest account), or complex system.

The native Win32 implementation that was developed shows that current computer hardware has enough performance to provide such functionality.

Acknowledgments

This work has partly been conducted in the context of the research project Embedded Interaction ('Eingebettete Interaktion') which is funded by the DFG ('Deutsche Forschungsgemeinschaft').

References

1. B. Shneiderman, Designing the User Interface, Addison Wesley, 1998, Third Edition.
2. B. L. Harrison, G. Kurtenbach, K. J. Vicente: An experimental evaluation of transparent user interface tools and information content. Proc. of the 8th ACM symposium on User interface and software technology. Nov 95, Pittsburgh, p.81-90.
3. P. Baudisch, C. Gutwin: Multiblending: displaying overlapping window simultaneously without the drawbacks of alpha blending. Proc. of CHI 2004, April 2004, pp. 367-374.
4. WYSIWYG tool tip project and software download page: http://www.hcilab.org/projects/tooltips/tooltips.htm
5. Microsoft Developer Network: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/windowing/hooks/abouthooks.asp