Stencil-Based Help and Tutorials

Caitlin Kelleher, Clifton Forlines, and Randy Pausch
Computer Science Department, Carnegie Mellon University
5000 Forbes Ave Pittsburgh, PA 15213
Email: caitlin+@cs.cmu.edu, forlines@cs.cmu.edu, pausch@cmu.edu

ABSTRACT

Users of traditional tutorials and help systems often have difficulty finding the components described or pictured in the documentation. Users also unintentionally miss steps, and perform actions that the documentation's authors did not intend, moving the application into an unknown state. We introduce an interaction technique that uses translucent colored stencils containing holes that direct the user's attention to the correct interface component and prevent the user from interacting with other components. Sticky notes on the stencil's surface provide necessary help and tutorial material in the context of the application. We have implemented a Java Swing prototype that allows stencils to be built around components, rather than screen pixels, so the stencils will work at multiple resolutions and automatically adapt to minor layout changes in the user interface. A simple authoring tool allows authors of tutorials and help instructions to concentrate on explaining the steps rather than describing how to find particular interface components.

KEYWORDS: Tutorials, help systems, interaction technique, transparent overlay, user interface design.

INTRODUCTION AND MOTIVATION

Software applications commonly provide either paper or online tutorials and reference manuals. This documentation is composed of sequences of written instructions and images that illustrate the steps a user should perform to accomplish specific tasks. The presentation of these instructions is often not ideal for users who are either familiarizing themselves with a new application or learning how to use new features within a familiar application.

Many users have difficulty locating the interface components described or pictured in the documentation [7]. Online documentation creates an additional problem: there are two versions of the component on the screen, real and pictorial. Users will often click on the image of a component in the tutorial rather than on the "real" component in the interface [7]. These difficulties are caused by the separation between the documentation and the application.

Paper and online instructions are usually presented statically and do not attempt to draw the user's eye to the current step's instructions. Because all instructions for a task are equally visually appealing, users often lose their place in the instructions while switching between the instructions and the application. Based on our observations, users often inadvertently miss steps and make mistakes.

Skipped steps, mistakes, and unspecified actions can hamper the documentation's ability to teach by moving the application into an arbitrary state. Because the user can put the application into an unintended state, and writing documentation for all possible states is prohibitively costly, users can find themselves in confusing application states that no longer match the next step of the documentation.

In addition to the problems discussed above, printed documentation often gets lost quickly. Although less likely to be lost, online documentation consumes valuable screen space and often requires the user to manage multiple windows.

In this paper, we present an interaction technique for presenting tutorial and help instructions that, in each step, draws the user's eye to the correct screen component and prevents the user from interacting with other components, allowing the user to concentrate on learning the application or feature (see Figure 1).

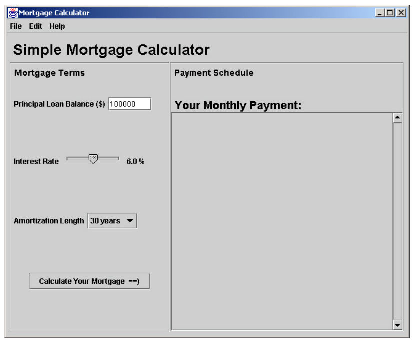
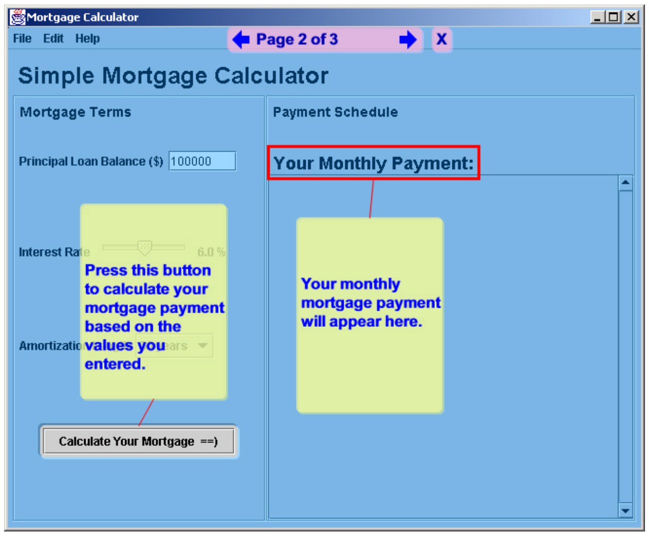

Mortgage Calculator
File Edit Help
Simple Mortgage Calculator
Mortgage Terms
Payment Schedule
Principal Loan Balance ($) 100000
Your Monthly Payment:
Interest Rate 6.0 %
Amortization Length 30 years
Calculate Your Mortgage ==)

Figure 1: **Before**- a simple mortgage calculator without a stencil (left) **After**- the calculator with a stencil containing a hole over the "calculate mortgage" button and a frame over the monthly payment (below).



Press this button to calculate your mortgage payment based on the values you entered.

Your monthly mortgage payment will appear here.

For the remainder of this paper, we will use the term "help" to mean both help instructions and tutorials. While we recognize that there are important differences between help systems and tutorials, we believe the stencils technique works well for presenting both types of instructions.

RELATED WORK AND APPROACH

Previous approaches to improving the presentation of help have primarily examined either helping the user determine what to do or preventing unanticipated application states from destroying the documentation's ability to teach.

Coachmarks[7] are markings, typically a circle, cross or check in red or green, drawn over a component in the interface to attract the user's attention to the component relevant to the current step. Sukaviriya and Foley [13] perform the sequence of steps with animated mouse cursors and keyboards to indicate user actions. Like Coachmarks, their technique shows the user what interface components to focus on; however, users may not fully understand what actions are necessary to accomplish a given task. Microsoft products include a similar "show me" help feature that performs steps for the user without showing the user how the steps were performed. None of these approaches prevents the user from putting the application into a state that the help's author did not anticipate.

Carroll and Carrithers [5] and Nymeyer [8] both restrict the options available to users. Carroll and Carrithers created a training version of an application. When users choose an advanced feature in the training version, the system responds with a dialog box stating that the chosen command is not available in the training system. The drawback to this approach is that users cannot determine in advance which menu items are available. Nymeyer visually disables components in the interface, changing the appearance of the application. If used in a tutorial setting, this might be surprising to the user when they repeat a task outside the context of the help system. Some tutorials also explicitly set the state of the application after each step. Using small steps, these tutorials prevent the user's explorations and mistakes from affecting more than a single step. However, users may not realize when they have made a mistake.

We present an interaction technique for presenting help based on sequences of full-screen, colored, transparent stencils containing holes. These stencils lie on top of the active application interface and intercept mouse and keyboard events. If an event occurs over a hole in the stencil, the event is passed to the GUI component beneath the hole. This prevents users from interacting with components that are covered by the stencil. Additionally, the holes in the stencil draw the user's eye to the component they should interact with during a given step. Notes on top of the stencil can supply additional information.

We have implemented a Java Swing-based prototype and Java interface that will allow other applications to build on our prototype. Stencil objects are stored using component-based information, so stencils will work at multiple resolutions and can adapt to changes in the layout of the interface.

The Need for Transparency

The stencils technique, without transparency, would probably be more of a hindrance than a help to a user trying to learn an application or feature; the user needs to see the context of the application while learning. Previous work has examined the use of transparency in interfaces and interaction techniques. Bartlett explored placing menus in the context of the application and using transparency to avoid obscuring the focal interface component [1]. Toolglass provides transparent tools representing commands that can be positioned over an object to select it. Clicking the tool applies the command to the selected object [3]. Magic lenses are movable filters that can provide an alternate view of data below the filter [12]. These techniques are summarized in a taxonomy by Bier et al [2]. None of these techniques have examined the possible uses of full-screen transparency.

INTERACTION DESCRIPTION

We present users with a series of stencils that guide them through a task step by step. In each step, users can interact only with components that have holes over them, preventing users from activating components that are not used in the current step. These restrictions help the user stay on the author's intended course. Instructions and additional information are presented using sticky notes on the stencil's surface. To aid readability, these notes are less transparent than the stencil, but the user can reposition them to get a better view of a part of the underlying interface. Users press "next" and "previous" arrows on the surface of the stencil to move through the sequence of steps. When a user presses the "previous" arrow the application must undo all actions associated with the current and previous stencils.

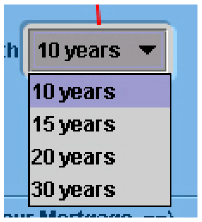


Figure 2: When clicked, pop-up menus appear on top of stencils.

The stencils technique has several advantages over previous work in help presentation. Stencils greatly decrease the number and types of mistakes that a user can make. The visual representation of the stencil draws the user's eye to the correct component for the current step. Each stencil provides a visual indication of what the user can do in that step, without altering the appearance of the application below. The stencils technique handles more complex interactions: pop-up menus appear on the on top of the stencil (see Figure 2); interface components can also be dragged from one hole to another. Instructions for each step are superimposed on the interface and displayed a single step at a time. Consequently, users cannot lose their place in the instructions or inadvertently skip steps. Stencils, like undo, may decrease the stress on users learning a new application or feature by giving the users confidence that they cannot make a mistake.

DESIGN ISSUES

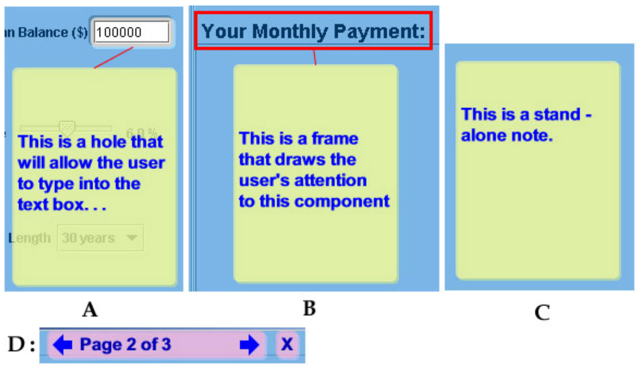
One possible problem with providing help and tutorial information in-context is that users may confuse interface components belonging to the help and tutorial information with those that are part of the application. To prevent this, interface elements associated with the tutorial have a different look than standard GUI elements, always appear on top of the stencils, and are slightly transparent so the user can see components in the underlying interface. It is always clear which interface elements belong to the help and which ones belong to the application interface. We have created four types of stencil objects for use in creating help.

Holes with attached notes are the most common interface elements. They provide a hole through which the user can interact with the underlying application component and an associated note that the author of the help can use to provide necessary information. We draw a red line connecting the note and associated hole (see Figure 3A).

Frames with attached notes highlight a particular application component without allowing the user to interact with it. They are typically used to bring aspects of the interface to the user's attention. For example, a frame could point out the results of a step the user has completed. An attached note provides any necessary explanation (see Figure 3B).

Stand-alone notes are used to provide a motivation or describe a goal that will take more than a single step. They are associated only with the stencil, not with any particular element in the application interface (see Figure 3C).

Navigation bars are automatically added to every stencil. They provide "next" and "previous" arrows. The navigation bar also indicates which step the user is currently performing and tells the total number of steps in the current task (see Figure 3D). An exit button allows users to close the stencil at any point.



in Balance ($)
100000
This is a hole that will allow the user to type into the text box...
Length
30 years
Your Monthly Payment:
This is a frame that draws the user's attention to this component
This is a stand - alone note.
A
B
C
D:
Page 2 of 3
X

Figure 3: Stencil objects. A is a hole with an attached note, B shows a frame with an attached note, C shows a stand alone note, and D is a navigation bar with previous, next, and close buttons.

Stencils are stored on a component rather than pixel basis to facilitate use at different screen resolutions and interface layouts. Consequently, all screen objects must be clearly associated with an interface component. While it is natural to want to draw directly onto the stencil while authoring help, drawn strokes often contain implicit references to components in the underlying interface. In early experimentation mocking up tutorials, we noticed that drawing on the surface is often used either to visually tie instructions to a particular hole or to draw attention to a component without allowing the user to interact with it. The four stencil objects allow authors of help to group instructions and components and draw the user's attention to interface components without allowing the user to interact with them, providing the benefits of drawing without the dangers.

IMPLEMENTATION

Our prototype implementation of stencils is written using the Java Swing framework. It uses the *glassPane* component in *JRootPane* to draw the stencil over the existing interface and intercept all mouse events. Each stencil maintains a list of holes and components associated with those holes. If a mouse event occurs inside a hole, the stencil passes the event to the component below; otherwise the stencil processes the event. Keyboard events are also controlled by explicitly managing which components in the underlying application have keyboard focus. Keyboard events reach a component in the application interface only if that component is associated with a hole that has the stencil's focus. A focus listener for the stencil's focused object prevents the user from moving to another component using the keyboard. If the layout of the application changes, the application can trigger a stencil update.

We have created a simple authoring tool for building help stencils. The authoring tool runs on top of the active application. Objects are added to the stencil by double clicking on its surface. By default, this creates a hole with an attached note. A right click menu allows authors to create a frame with a note or a stand-alone note rather than a hole with a note. The author can reposition notes by dragging them on the surface of the stencil and add information by typing. Notes are visually attached to their associated holes or frames with a line that updates when they are moved.

This style of authoring help allows the author to focus on the explanations of the steps rather than descriptions to the users of how to find the correct interface element on the screen. We believe that this simple authoring tool will allow teachers to easily build special purpose help stencils for their students and allow advanced users to create help stencils in response to questions they receive from other users.

DISCUSSION AND FUTURE WORK

Our future work lies in three areas: evaluating stencils as a method for presenting help, experimenting with the design of stencils and stencil objects, and expanding the technique.

Evaluation

We intend to perform a user study similar to Booher [4] and Palmiter et al [10] comparing stencils with other techniques for presenting help including traditional paper, on-line tutorials and Coachmarks. We will compare these techniques using time to completion, error rate, and user confidence. One risk of the stencils technique is that it may lead to a negative training effect in which users go through the help materials quickly without understanding each step. To evaluate this possibility, we will study how users remember material presented in the different formats.

Design
We currently present all stencils with uniform transparency. A study on transparent layered interfaces [6] found that users have different preferences for the levels of transparency that best allow them to focus on each interface layer. One enhancement would be to provide control over the degree of transparency of stencils and notes. Olsen et al explored the use of highlighting and blending to emphasize particular interface elements and de-emphasize the remaining elements [9]. It might be beneficial to allow the stencil author to similarly de-emphasize parts of the interface using non-uniform transparency.

We chose blue stencils based on user preference in informal pilot tests. However, stencils need to draw the user's eye to components with holes over them without negatively impacting the readability of the underlying interface. Stencil colors could be automatically chosen, perhaps by maximizing contrast between stencils and application components.

In addition, we would like to allow organizations to customize the look of their own stencil objects. This would allow businesses to customize help material for their own use while maintaining a clear difference between stencils authored by the makers of the software and stencils authored by someone in the local office.

Expanding Stencils
The current prototype implementation of stencils does not include a concept of state. Consequently, while stencils can greatly reduce the numbers and kinds of errors possible, they cannot prevent mistakes entirely. Incorporating a notion of state would allow a user to move to the next step only if the current step had been performed correctly.

Since we know where the user's focus should be directed, we could use animation to show the user the sequence of steps. The border of the hole where a drag should begin could gently animate to draw the user's eye. Then, once the user begins the drag operation, the corresponding border of the drop target would begin animating. An author of help may also want stencils to automatically advance to the next stencil when the user's action will cause a dramatic change in the application's appearance, such as the disappearance of a component used in the current step.

While the stencils technique seems ideal for users who are attempting a particular task for the first time, it seems less well suited to users who have performed a task before and just want a quick reminder. We plan to explore the possibility of allowing users to make stencils less restrictive; for example, a more experienced user might request unrestricted access to a particular sub-region of the screen. We can also generate traditional online tutorials from the stencils by capturing images of the components and presenting them alongside the text in the notes. A user who only wanted clarification on the steps for a given task might prefer a less restrictive presentation method.

Finally, we would like to investigate the possibility of applying these stencils to stroke-based interfaces by providing guide strokes on the stencils. A user going through the help stencils could then "trace" the guide strokes to learn how to use a stroke-based technique.

ACKNOWLEDGMENTS
The authors would like to thank our colleagues in the Squeak Research Group (now at Viewpoints Research Institute) for providing a rich research environment in which to develop the idea of stencils and our colleagues in the Stage 3 Research Group at Carnegie Mellon for their ideas and support as we implemented and experimented with stencils.

REFERENCES

1. Bartlett, J., "Transparent Controls for Interactive Graphics" Palo Alto, California, 1992, Western Research Laboratory**:** 9.
2. Bier, E., M. Stone, K. Fishkin, and W. Buxton. "A Taxonomy of See-Through Tools," *Proceedings of CHI 1994*, Boston, Massachusetts.
3. Bier, E., M. Stone, K. Pier and W. Buxton. "Toolglass and magic lenses: the see-through interface," *Proceedings of International Conference on Computer Graphics and Interactive Techniques 1993*.
4. Booher, H. R. "Relative Comprehensibility of Pictorial Information and Printed Words in Proceduralized Instructions." *Human Factors* 17**,**3 (1975).
5. Carroll, J. and C. Carrithers. "Training Wheels in a User Interface." *Communications of the ACM* 27**,**8 (1984): 800-806.
6. Harrison, B. L., H. Ishii, K. J. Vicente and W. Buxton. "An Evaluation of a Display Design to Enhance Focused and Divided Attention," *Proceedings CHI 1995*, Denver, Colorado.
7. Knabe, K. "Apple Guide: a case study in user-aided design of online help," *Proceedings of CHI 1995*, Denver, Colorado.
8. Nymeyer, A. "A Grammatical Specification of Human-Computer Dialogue." *Computer Languages* 21**,**1 (1995):1-16.
9. Olsen, D., D. Boyarski, T. Verrati, M. Phelps, J. Moffett, and E. Lo, "Generalized Pointing: Enabling Multiagent Interaction," *Proceedings of CHI 1998*, Los Angeles, California.
10. Palmiter, S., and J. Elkerton, "An Evaluation of Animated Demonstrations for Learning Computer-based Tasks," *Proceedings of CHI 1991*, New Orleans, Louisiana.
11. Selber, S. A., Johnson-Eilola, J, Mehlenbacher, B, "Online Support Systems: Tutorials, Documentation, and Help" in The Computer Science and Engineering Handbook. J. A. B. Tucker. Boca Raton, FL, CRC P**:** 1619-1643 1997.
12. Stone, M., K. Fishkin and E. Bier. "The Movable Filter as a User Interface Tool," *Proceedings of CHI 1994*, Boston, Massachusetts.
13. Sukaviriyam, P. and J. Foley. "Coupling a UI framework with automatic generation of context-sensitive animated help," *Proceedings of UIST 1990*, Snowbird, Utah.