User Interface Designers, Slaves of Fashion

Jef Raskin, Consultant

The discipline, science, and art of interface design has gone stagnant. The most widely read books on the subject are primarily compendia of how to make the best of received widgets. The status quo is mistaken for necessity. Constrained in this chamber pot, designers wander around giving the users of their products little comfort or fresh air.

For example: I have before me a collection of books about interface design, written by acknowledged experts in the field, and they contain many good ideas, effective heuristics, and describe traps to avoid what all too many interfaces, especially Web interfaces, fall into. Not one of them, however, points out that the cut-and-paste method of moving text is inherently faulty.

Rare are the computer users who will not admit that they have committed the crime of “cut with intent to paste” and then become distracted (perhaps by a telephone call, more commonly by seeing something else they want to move). They then make a second cut and, with luck, succeed in pasting that item where it belongs. Meanwhile, the first cut has been lost. Some systems may have methods for retrieving it, but that is not the point. The concept of cut-and-paste causes problems. If methods exist to fix these problems, then you must learn those methods; they are merely Band-Aids over festering wounds. It is better to prevent injury in the first place.

The reasons why cut-and-paste causes problems are rather obvious: You can’t see what has been cut; it is in some invisible hideaway (“buffer” to the cognoscenti) where it can be easily forgotten. A number of actions, such as closing the document or making another cut, cause the cut material to disappear as a side effect; furthermore, there is a delay between cutting and pasting during which you have to perform a visual search (or worse, a computer-mediated search) to move the cursor to the desired paste site. A cognitive psychologist might point out another reason for the cut-and-paste problems: All that mental and physical activity usually takes longer than the decay of the short-term memory of having made the cut, but we can be less fancy and say that in the hustle of editing we sometimes lose track of the fact that we’ve cut something.

**The status quo
prevails IN INTERFACE
DESIGN, AND THE
FLAWED CONCEPT OF
CUT-AND-PASTE IS
A PERFECT EXAMPLE.**

An often-proposed fix available in some software is select-and-drag. In this method you start, as before, by selecting some text. Instead of cutting it, however, you do some-

thing far safer: You point to it with the mouse, and while holding down a button move the mouse to the location where the text is to be inserted. The text, which in the best implementations remains visible and readable during this process, moves with the cursor to the new location and—when you let go of the button—adheres there. With this method, you do not lose text, although sometimes you drop it in the wrong place.

Unfortunately, if the location to which you wish to move the text is off-screen, you may have to use some other mechanism to find the target location. In such cases, you are back to cut-and-paste. Instead of fixing the problem itself, the designers have added a new mechanism that only sometimes fixes the problem. This kind of “solution” adds to the software bloat we all hate. It is a design method we see all too often: A mechanism fails in certain cases, so instead of seeing if we can redesign the mechanism so that it always works correctly, we add a new mechanism to be used in those cases where the original failed. Now we have two mechanisms, both of which are damaged goods. The user manual (online, so the user has to learn how to run the manual-reading software) has gotten longer, the software more complex, and there are more opportunities for the code to have bugs. As is always the case with larger software, programming time has increased, and there is more software to maintain and document. In short, everything we know about making software better has been frustrated and we haven’t solved the original problem.

In fact, we have created new opportunities for annoying the user; dragging text brings with it its own pitfalls. The most glaring one is that you cannot easily change the selection you wish to move. Here’s what happens: You select some text. Then you realize that you have not selected exactly the right text (you may wish to move

fewer or more characters, for example). Let's take the case where you decide to move only a few words from a longer phrase you thought you wanted to move. You place the cursor at one end of the desired sub-phrase and try to drag the mouse (push the appropriate mouse button and move the mouse) across the words to select them. You do not get the desired selection because this action causes the already-selected text to move. The fault lies in using the same mechanism for both selecting and moving text. In this case, you mean one thing by your gesture but the computer dictates a different meaning. This is a classic mode error.

Human beings cannot pay attention to their task and to system state at the same time. Mode errors are caused by the system responding differently to a particular user gesture when the system is in different internal states. Interface designers have known about and published papers about the difficulties caused by modes since at least the early 1970s. Nonetheless, they continue to design modes into interfaces and thus continue to ill-serve users. There is much more to be said about modes and why you can't solve the problem with mode indicators, but there is not room here to discuss the subject (I wrote about it in detail in my book, *The Humane Interface*, Addison-Wesley, 2000).

A very few systems avoid the entire issue by using a "move" command. You make a selection, which per-

sists (and so remains visible and in its place) while you move the cursor to the target location. You then issue the move command (which requires no more effort than that required to issue the paste command) and the selection moves. Not only have we eliminated one command (before you had to issue both cut and paste commands; now you issue one move command), but the text you are moving never vanishes; and if you start a second move before completing the first, nothing is lost.

That such a simple solution exists, has been available for decades, yet is rarely used shows how interface designers have become slaves of fashion and how corralled they are by convention. Q

LOVE IT, HATE IT? LET US KNOW:

queue-ed@acm.org or www.acmqueue.org/forums

JEF RASKIN is best known for his book, *The Humane Interface* (Addison-Wesley, 2000) and for having created the Macintosh project at Apple. He holds many interface patents, consults for companies around the world, and is often called upon as a speaker at conferences, seminars, and universities. His current project, *The Humane Environment* (<http://humane.sourceforge.net/home/index.html>), is attracting interest both in the computer science and business worlds.

© 2003 ACM 1542-7730/03/0900 \$5.00.

acm
queue
tomorrow's computing today

Coming in October

The Future of Power
With power consumption outstripping Moore's Law, and the latest chips hot enough to cook on, what does the future hold?

- The inevitability of reconfigurable systems
- Understanding power management and ACPI
- Supercomputing—It's not just about megahertz
- Plus: Code reading, why should you do it?